Titre: Title:	IRIS : amélioration d'une méthode de génération de colonnes pour la confection d'horaires d'infirmières
Auteur: Author:	Pascal Labit
Date:	2000
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Labit, P. (2000). IRIS : amélioration d'une méthode de génération de colonnes pour la confection d'horaires d'infirmières [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/8610/

UNIVERSITÉ DE MONTRÉAL

IRIS : amélioration d'une méthode de génération de colonnes pour la confection d'horaires d'infirmières

Pascal Labit
DÉPARTEMENT DE GÉNIE ÉLECTRIQUE ET DE GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRE ÈS SCIENCES APPLIQUÉES (M.Sc.A.)
(GÉNIE ÉLECTRIQUE)
décembre 2000

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE

Ce mémoire intitulé:

IRIS : amélioration d'une méthode de génération de colonnes pour la confection d'horaires d'infirmières

présenté par: Pascal Labit

en vue de l'obtention du diplôme de: Maître ès science appliquées

a été dûment accepté par le jury d'examen constitué de:

Pesant Gilles, Ph. D., président

Jaumard Brigitte, Thèse de doctorat, Thèse d'habilitation, membre et directeur de recherche

Gendron Bernard, Ph. D., membre

Remerciements

Je remercie Brigitte Jaumard pour m'avoir proposé ce projet de maîtrise enrichissant, ainsi que pour m'avoir laissé une totale liberté dans la poursuite de l'aboutissement de cette maîtrise.

Je tiens à remercier tout particulièrement Tsévi Vovor, avec qui j'ai eu plaisir à travailler et au contact de qui j'ai appris énormément.

Mes remerciements particuliers vont aux personnes des hôpitaux avec qui j'ai collaboré durant le développement de l'outil logiciel. Ils ont su montrer patience et compréhension pendant les différentes phases du développement. Merci à Carolyn Dagenais, à Gianna Lepiane et à Marie-France Noëlle de l'hôpital Royal Victoria et à Réal Burke du CHUM informatique.

Je remercie également Daniel Villeneuve pour son aide dans la compréhension du logiciel Gencol et dans la tentative d'unification de nos deux modèles.

Je remercie enfin bien évidemment tous les collègues et amis passés ou présents du GERAD avec qui j'ai eu tant de plaisir à collaborer.

Résumé

Générer des horaires de manière automatique est un sujet très actuel et de plus en plus considéré par les gestionnaires. L'automatisation de cette tâche permet des gains de ressources considérables, tout en permettant la prise en compte dans la plupart des cas de plus de préférences des différents intervenants.

Le problème de génération automatisée d'horaires pour des infirmières étudié ici fait appel à la méthode de génération de colonnes pour la résolution. Un modèle mathématique développé précédemment a pour ce faire été adapté puis vérifié de manière théorique. Ce modèle est basé sur un algorithme de plus courts chemins dans un graphe acyclique avec contraintes de ressources. Diverses améliorations concernant notamment la gestion des ressources sont proposées. Les résultats obtenus ont permis de valider le modèle mathématique et le programme IRIS de résolution associé avec les données de deux unités de soins de l'hôpital royal Victoria (centre des naissances, unité de dialyses). Une implantation du programme IRIS est en cours de planification dans l'ensemble des unités du MUHC (McGill University Health Center) qui comprend cinq hôpitaux dont l'hôpital Royal Victoria.

Mots Clefs : génération d'horaires d'infirmières, optimisation, génération de colonnes, plus courts chemins.

Abstract

Automated generation of scheduling is a topic of growing interest for managers of human resources. The automation of the generation of schedules allows quite significant resource gains, while making easier the decision making process even with the expressions of staff preferences.

The problem of automated generation of schedules which is studied here considers a column generation mathematical model. It was first proposed in earlier works of Jaumard, Semet and Vovor (1998), and in this master thesis has been extended and validated for several resource variants. The model consists in a decomposition of the initial problem in a master and an auxiliary subproblems, the last one corresponding to a shortest path problem in an acyclic graph with several resource constraints. Various improvements have been proposed. Experimental results allows a validation of both the mathematical model and the associated program IRIS on two data sets from the Royal Victoria Hospital (Birth Center and Dialysis unit). An implementation of the IRIS program is now planned in all the units of the MUHC (McGill University Health Center) which includes five hospitals including the Royal Victoria Hospital.

Key-words: nurse schedules generation, optimisation, column generation techniques, shortest path algorithm.

Table des matières

Liste des tableaux

Liste des figures

Liste des annexes

Introduction

La génération automatisée d'horaires de personnel est un sujet très actuel et de nombreux auteurs s'y intéressent. Que ce soit des équipes de personnels navigants dans les compagnies d'aviation ou des équipes de personnels soignants, les domaines d'applications sont très nombreux. En particulier, les gestionnaires se sont aperçus que la puissance grandissante des ordinateurs et les méthodes de plus en plus raffinées en recherche opérationnelle permettaient de prendre en compte de plus en plus de contraintes exprimées par les personnels pour lesquels ces horaires sont générés. De plus, cette automatisation de la génération des horaires permet de libérer la ressource affectée habituellement à cette planification. Il faut en effet savoir que, dans le cas qui nous préoccupe dans ce mémoire, c'est-à-dire la génération d'horaires pour du personnel infirmier, la planification des horaires pour un horizon de six semaines et une équipe de 80 personnes représente une semaine de travail pour la personne responsable.

Comme nous le verrons dans ce mémoire, ce problème a déjà été abordé par plusieurs auteurs. Nous nous concentrerons plus particulièrement sur la génération automatisée d'horaires pour des équipes d'infirmières. Ce problème précis avait déjà été traité par Jaumard, Semet et Vovor [1] puis repris par Vovor dans le cadre de sa thèse de doctorat [2]. Cependant, de nombreuses améliorations ont été apportées.

Nous avons d'abord apporté une preuve théorique complète et originale à l'exactitude de la gestion des ressources sur les graphes utilisés par le problème

auxiliaire. En effet, une des questions posées était de savoir si les ressources modélisaient bien tous les cas particuliers que l'on voulait voir modélisés. T. Vovor avait déjà fait cette étude [2], mais des modifications dans la modélisation (incertitudes résolues en ce qui concerne le rapport congés et charge de travail, simplifications apportées dans la modélisation de la charge de travail, amélioration de la ressource reliée aux fins de semaine) nous ont forçé à la reprendre. Nous avons ensuite apporté des améliorations sur les aspects suivants :

- modélisation des jours de repos versus jours de congés avec une nouvelle modélisation de la ressource gérant les jours de congés;
- introduction d'une procédure d'arrondi pour accélérer la génération de solutions réalisables;
- écriture de développements théoriques pour prendre en compte des contraintes additionnelles : contrôle du nombre de jours successifs travaillés, respect des règles de la convention collective quant aux prises de jours de congés;
- étude de l'introduction de profils d'infirmières.

Dans ce mémoire, l'organisation des chapitres va suivre la démarche intellectuelle que nous avons suivi lors du développement du logiciel de génération d'horaires pour des infirmières. Nous commencerons donc par rappeller les grandes lignes de la méthode utilisée pour résoudre le problème de génération d'horaires pour des infirmières (voir chapitre 1), tout en présentant les particularités et autres améliorations que nous lui avons apportées.

La méthode utilisée, dite de génération de colonnes, est basée sur la méthode révisée du simplexe. Comme nous le verrons dans le chapitre 2, le problème auxiliaire à résoudre pour trouver les colonnes entrantes est un problème de plus courts chemins dans un graphe avec étiquettes de ressources sur les sommets et les arcs. Nous avons accordé une grande importance à la vérification de la construction du

graphe correspondant au problème auxiliaire. Nous avons donc réalisé une étude théorique originale qui passe en revue tous les cas de figures pouvant se présenter lors de la construction du graphe. Cette étude constitue le corps du chapitre 2. Nous présenterons, pour chaque ressource définie dans le problème auxiliaire, un graphe qui servira d'exemple et de justification à la méthode de résolution adoptée.

Lors du développement, de nombreuses améliorations ont été apportées. Nous décrirons (chapitre 3) celles qui l'ont été au sein du modèle, notamment la modélisation de la quatrième ressource. Nous verrons ensuite comment plusieurs améliorations successives ont permis de réduire les temps de calcul. Enfin, dans un dernier chapitre (chapitre 4), nous présenterons les résultats obtenus et montrerons comment ces résultats répondent aux exigences de nos clients et se comparent à leurs horaires. Nous terminons sur les différentes perspectives quant à l'avenir du projet.

Chapitre 1

Problématique et méthode utilisée

L'objet de ce chapitre est de présenter le problème de génération automatisée d'horaires d'infirmières. Nous présentons d'abord la modélisation mathématique adoptée pour formaliser ce problème. Ensuite, nous présentons la méthode de résolution utilisée qui consiste à considérer une décomposition du problème initial en un problème maître et un problème auxiliaire. Un chapitre ultérieur (chapitre 2) présentera de manière beaucoup plus détaillée le problème auxiliaire.

1.1 Présentation du problème de confection d'horaires d'infirmières

Générer de manière automatisée des horaires d'infirmières est un problème pouvant paraître de prime abord facile à résoudre. Néanmoins, les différentes contraintes intervenant rendent le problème particulièrement difficile. Il faut savoir, à titre d'information, que la génération d'un horaire pour une équipe de 80 infirmières sur un horizon de six semaines monopolise l'équivalent d'une infirmière-

chef à temps plein pendant une semaine de travail. Il y a donc un intérêt à disposer d'un outil générant ces horaires automatiquement, l'infirmière-chef n'ayant plus qu'à contrôler dans un premier temps la validité des horaires générés. Cette section va donc nous servir à présenter la problématique qui nous a été soumise.

1.1.1 Objectif

L'objectif du problème posé est la conception et le développement d'une méthode d'optimisation pour la génération automatique d'horaires d'infirmières pour une unité de soins donnée. La solution finale devra contenir un horaire pour chaque infirmière présente dans l'unité considérée, si un horaire réalisable existe. Ces horaires pris ensemble devront satisfaire à des contraintes de groupes qui auront été spécifiées. S'il n'existe pas d'horaire réalisable, alors le modèle devra traiter les différents cas de non réalisabilité (section 1.4.3).

1.1.2 Principales contraintes

Comme on l'a vu plus haut, la solution trouvée doit satisfaire plusieurs contraintes. Les premières sont globales. Elles concernent les horaires pris dans leur ensemble : ces contraintes stipulent que pour chaque jour, pour chaque période de travail, on doit satisfaire un certain quota (un certain nombre d'infirmières avec un niveau donné de qualification).

Le second type de contraintes, les plus nombreuses, sont les contraintes individuelles exprimées pour chaque infirmière. Ces contraintes seront transférées dans le problème auxiliaire. Elles peuvent être divisées en deux catégories.

- Les contraintes dures, qui doivent être obligatoirement satisfaites dans chaque horaire généré pour l'infirmière considérée. Ces contraintes sont au nombre de quatre principales : la charge de travail, les rotations entre fins de semaine travaillées et non-travaillées, les rotations entre les différents

types de quart affectés et les congés accordés à l'infirmière dans l'horizon considéré. On notera que les contraintes concernant les fins de semaine et les congés sont optionnelles. Il faut également ajouter à ces contraintes dures des pré-affectations et affectations interdites qui sont spécifiées à l'avance pour chaque infirmière.

- Les contraintes dites "molles" (*soft* en anglais), qui sont prises en compte sous la forme de bonus ou de pénalités associés au respect ou non respect de ces contraintes lors de l'évaluation de la qualité de l'horaire généré. Ces contraintes couvrent plusieurs champs d'application :
  - les ratios entre les différents types de quarts de travail (jour, soir ou nuit) pour obtenir un horaire plus ou moins équilibré;
  - les préférences et aversions exprimées par l'infirmière.

1.2 Méthodes existantes pour résoudre le problème

Comme vu plus haut, la génération automatisée d'horaires est un problème très populaire, du fait au moins du temps et des ressources qu'il permet d'économiser. Il est donc logique que plusieurs auteurs s'y soient intéressés et que de nombreuses méthodes aient été testées pour résoudre le problème.

Dans cette section, nous allons passer en revue les principales méthodes qu'une revue de la littérature a permis de mettre en évidence.

Il faut savoir que pour résoudre ce genre de problèmes de grande taille, il existe deux écoles :

- La première considère que les différents intervenants ne sont pas forcément intéressés par la solution optimale du problème à résoudre. Une solution réalisable de bonne qualité peut souvent suffire. Dans ce cas, les auteurs ont

étudié des méthodes dites heuristiques, c'est-à-dire qui ne garantissent pas de trouver la solution optimale.

- La seconde considère que l'optimalité de la solution trouvée est importante. Dans ce cas, une méthode dite exacte sera utilisée pour s'assurer que la meilleure solution trouvée est bien la solution optimale du problème considéré. Dans de nombreux cas, il peut être utile de disposer de cette information. Notamment à cause de la puissance de calcul et des capacités sans cesse grandissantes des machines, il est intéressant de chercher une solution optimale à un problème. Ceci peut être utile pour s'assurer de la qualité d'un horaire répondant à des critères donnés. Par exemple, seule une méthode exacte permet de chercher la solution optimale à un problème pour ensuite étudier combien les solution réalisables trouvées pendant l'optimisation sont loin de l'optimum.

Nous allons donc lister dans les sections suivantes les différentes méthodes existantes pour résoudre le problème, que ce soit de manière exacte ou non. Une revue des méthodes existantes pour résoudre ce problème avait été proposée par Hung [3].

Une approche classique utilisée pendant de nombreuses années a été de développer des méthodes had-oc basées sur l'expertise humaine et l'utilisation de tableurs [4]. Pourtant, les années 70 ont également vu l'apparition de méthodes plus novatrices basées sur la recherche opérationnelle [5], [6]. Nous allons dans les paragraphes suivants détailler plus avant certaines méthodes héritées de ces travaux.

1.2.1 Méthodes exactes

Curieusement, peu d'auteurs ont fait appel à une méthode exacte pour résoudre le problème de génération automatisée d'horaires pour des infirmières. Un modèle,

ancêtre du modèle développé présentement avait été proposé par Jaumard, Semet et Vovor ([1],[2]). Ce modèle fait appel à la programmation linéaire pour résoudre le problème de génération des horaires. C'est à partir de ce modèle que nous avons travaillé, de nombreuses modifications et raffinements ayant été apportés par la suite.

Ce modèle constitue par ailleurs le seul exemple trouvé de méthode exacte appliquée à ce problème particulier de génération automatisée d'horaires d'infirmières.

1.2.2 Méthodes heuristiques

Par contre, de nombreux auteurs se sont concentrés sur une résolution heuristique du problème, prenant comme objectif de générer le meilleur horaire possible mais sans s'assurer que cet horaire soit le meilleur que l'on puisse générer, autrement dit la solution optimale du problème à résoudre.

Plusieurs équipes se sont plongées sur la résolution du problème en utilisant la programmation par contraintes. En effet, ce type de problème se prête très bien à ce traitement puisque la plus grande part des contraintes prises en compte lors de la résolution peuvent s'exprimer sous forme de contraintes logiques.

Abdennadher et Schlenker [7] ont utilisé un mélange de programmation par contraintes et d'intelligence artificielle pour trouver une solution à ce problème. Leur modèle nécessite l'intervention de l'utilisateur pour résoudre certains conflits qui peuvent être créés lors de l'exécution de l'outil basé sur cette méthode. Ici intervient une différence que l'on retrouve entre plusieurs modèles : certains modèles sont très orientés interactivité et vont demander la participation de l'utilisateur pour bâtir les horaires. D'autres modèles sont plus indépendants et ne nécessitent pas l'intervention de l'utilisateur.

L'équipe de Weil *et al.* [8] travaille également sur une modélisation du problème faisant appel à la programmation par contraintes. Cette modélisation a

d'ailleurs débouché sur un outil utilisé dans plusieurs hôpitaux français.

Il existe également une autre approche pour modéliser le problème de génération automatisée d'horaires pour des infirmières. Cette approche est utilisée par Dowsland [9]. Il s'agit d'une méthode basée sur une heuristique de type recherche tabou. Le principe de cette heuristique est bien connu mais il a été, dans ce modèle, raffiné pour en améliorer les performances. Il s'agit en fait d'osciller entre des phases durant lesquelles on cherche à améliorer la réalisabilité de la solution courante, et d'autres durant lesquelles on essaie de maximiser le respect des différentes préférences exprimées par les infirmières pour lesquelles on génère les différents horaires. Ce modèle permet de générer des horaires pour trois équipes d'infirmières simultanément pour un horizon de planification de six semaines. Il permet également de prendre en compte l'historique des affectations effectuées précédemment.

1.3 Modélisation retenue

Nous allons dans cette section proposer la modélisation qui a servi de base à l'outil développé pendant les travaux de recherche de ce mémoire de maîtrise. Cette modélisation avait été proposée par Jaumard, Semet et Vovor [1] (voir également la thèse de Vovor [2]), puis modifiée par Vovor et l'auteur pour répondre à un niveau de raffinement plus poussé dans la modélisation du problème. Nous présenterons donc, dans un premier temps, les données dont nous disposons pour résoudre le problème, puis la modélisation mathématique que nous avons retenue.

1.3.1 Problème maître

Notre problème utilisant une technique de résolution dite de génération de colonnes (voir section 1.4.2 et [10], [11]), il est décomposé en deux sous-problèmes distincts, que nous appelons problème maître et problème auxiliaire. Ces deux sous-

problèmes vont être décrits dans les deux sous-sections suivantes. Nous commençons par le problème maître.

1.3.1.1 Données du problème

Les hôpitaux avec lesquels nous travaillons pour générer des horaires d'infirmières fournissent les données décrites ci-après.

- Les infirmières, notées N_k. Chacune d'entre elles est associée à des caractéristiques et des contraintes intrinsèques (comme des disponibilités, des préférences personnelles, un niveau d'expérience, ...). On reviendra sur ces contraintes intrinsèques plus loin dans le chapitre. On notera dans la suite K l'ensemble des indices d'infirmières.
- L'horizon de planification, chaque jour étant noté D_i. Il s'agit ici de l'ensemble des jours pour lesquels on veut générer un horaire pour chaque infirmière. En général, les horizons ont une durée de quatre à six semaines. On notera I l'ensemble des indices relatifs aux jours de l'horizon courant (en général $I = 1, 2, .., 28$ ou $I = 1, 2, .., 42$.)
- Les quarts de travail que l'on peut affecter, S_j. Ces quarts sont les "briques de base" avec lesquelles on bâtit les horaires. Un quart de travail défini un nombre d'heures consécutives durant lesquelles l'infirmière doit être présente. On note J l'ensemble des indices de quarts de travail.
- Les contraintes de quotas, qui spécifient que l'on doit avoir Q_{ij} infirmières présentes le jour D_i pour effectuer le quart S_j. Ces quotas sont spécifiés avec une fenêtre de réalisabilité $[Q_{ij} - Q_{ij}^{max\_deficit}, Q_{ij} + Q_{ij}^{max\_surplus}]$. Des pénalités pour le dépassement (*overstaffing*) et le manque de personnel (*understaffing*) permettent de contrôler le respect des quotas au plus serré. Ces contraintes sont celles qui resteront dans le problème maître après la décomposition en deux sous-problèmes. Les contraintes intrinsèques au

contenu des horaires générés pour chaque infirmière seront quant à elles transférées dans le problème auxiliaire.

1.3.1.2 Variables du problème

Le but du problème va donc être de générer un horaire réalisable pour chaque infirmière de telle sorte que l'ensemble de ces horaires satisfasse les contraintes de quotas. La technique de génération de colonnes (section 1.4.2) nous permettra de générer un grand nombre d'horaires réalisables pour chaque infirmière N_k (notés H_{ks}, où s correspond au $s-ième$ horaire réalisable généré pour l'infirmière considérée, l'indice s décrivant un ensemble noté S), et affecterons à chacun un poids dépendant des préférences et de la contribution de cet horaire à la satisfaction des contraintes de quotas (on notera ce poids P_{ks}, voir la section 1.3.1.5 pour le détail du calcul du poids).

On définit de plus la variable auxiliaire suivante :

$$a_{ijks} = \begin{cases} 1 & \text{si le quart } S_j \text{ est affecté le jour } D_i \\ & \text{dans l'horaire } H_{ks}, \\ 0 & \text{sinon.} \end{cases}$$

Les variables du problème maître sont décrites ci-dessous.

- Les variables de décision y_{ks} définies comme suit :

$$y_{ks} = \begin{cases} 1 & \text{si l'horaire } H_{ks} \text{ (s-}ième \text{ horaire réalisable pour l'infirmière } N_k) \\ & \text{est conservé dans la solution optimale} \\ 0 & \text{sinon} \end{cases}$$

- Les variables d'écart pour les contraintes de quotas s_{ij}.

1.3.1.3 Contraintes du problème maître

Les contraintes du problème maître sont ici décrites.

- Le respect des contraintes de quotas :

$$\forall i \in I, \forall j \in J, \quad \sum_k \sum_s a_{ijks} \cdot y_{ks} - s_{ij} = Q_{ij}.$$

On associe à ces contraintes les variables duales λ_{ij}.

- Le respect de la fenêtre maximale de quota (qui donne les bornes sur les variables s_{ij}) :

$$\forall i \in I, \forall j \in J, \quad -Q_{ij}^{max\_deficit} \leq s_{ij} \leq Q_{ij}^{max\_surplus}.$$

- L'existence d'un et un seul horaire pour chaque infirmière :

$$\forall k \in K, \quad \sum_s y_{ks} = 1.$$

La variable duale associée à cette contrainte sera notée μ_k.

1.3.1.4 Formulation du problème maître

Le problème maître peut se formuler de la façon suivante:

$$\text{min.} \sum_k \sum_s P_{ks} \cdot y_{ks}$$

$$\text{s.l.c.} \begin{cases} \sum_k \sum_s a_{ijks} \cdot y_{ks} - s_{ij} = Q_{ij} & i \in I, j \in J \\ \sum_s y_{ks} = 1 & k \in K \\ -Q_{ij}^{max\_deficit} \leq s_{ij} \leq Q_{ij}^{max\_surplus} & i \in I, j \in J \\ y_{ks} \in \{0,1\} & k \in K, s \in S. \end{cases}$$

1.3.1.5 Détails du calcul du poids P_{ks} d'un horaire donné

Le calcul du poids affecté à un horaire H_{ks} dans la fonction objectif du problème maître se fait de la façon décrite ci-après. Dans cette section, nous adopterons les notations décrites ci-dessous.

- S_k désignera l'ancienneté de l'infirmière N_k. Cette ancienneté a été normalisée par rapport à l'ancienneté la plus grande parmi toutes les infirmières considérées.

- L'infirmière peut exprimer, pour chaque affectation potentielle, une préférence ou aversion que l'on notera f_{ij} (on aura $f_{ij} > 0$ pour une préférence et $f_{ij} < 0$ pour une aversion). Si pour une affectation donnée aucune préférence n'est exprimée, on aura par défaut $f_{ij} = 0$.
- De même que pour les préférences, le gestionnaire peut vouloir faire entrer en ligne de compte la composante salariale. Si tel est le cas, on note g_{ij} la composante salariale attribuée à chaque affectation potentielle.
- Une pénalité $p_{week-end}$ sera appliquée pour chaque fin de semaine brisée (fin de semaine durant laquelle un seul des deux jours est travaillé) présente dans l'horaire généré. On notera pour ce faire $\mathcal{N}_{broken\_week-ends}$ le nombre de fins de semaine brisées affectées à l'infirmière considérée.
- Une distance sera introduite pour respecter les contraintes de ratios. Pour chaque type t de quart de travail (t valant "jour", "soir" ou "nuit"), on dispose d'une fourchette $[min_t, max_t]$ dans laquelle on doit se situer. Pour le vérifier, on calcule :

$$ratio_t = \frac{\text{nombre de quarts affectés de type } t}{\text{nombre de quarts de travail affectés}} \times 100.$$

Dans le cas où l'on se situe en dehors de l'intervalle, on applique la pénalité suivante :

$$p_{ratio_t} = \begin{cases} 0 & \text{si } min_t \leq ratio_t \leq max_t \\ \frac{1}{100} \times |ratio_t - M_t| & \text{sinon.} \end{cases}$$

où

$$M_t = \begin{cases} min_t & \text{si } ratio_t < min_t, \\ max_t & \text{si } ratio_t > max_t\ . \end{cases}$$

Le coût P_{k_s} d'un horaire H_{k_s} est alors calculé de la façon suivante :

$$P_{k_s} = \sum_i \sum_j S_k . a_{ijk_s} . (f_{ij} + g_{ij}) + p_{week-end} \times \mathcal{N}_{broken\_week-ends} + \sum_t p_{ratio_t}$$

1.3.2 Coût réduit associé aux variables

Cette brève section va servir à définir le coût réduit tel que nous le prenons en compte dans notre problème.

Dans un algorithme de génération de colonnes, le but du problème auxiliaire est de générer la ou les meilleures colonnes potentiellement entrantes. Pour ce faire, comme lors d'un algorithme du simplexe classique, l'objectif du problème auxiliaire est de trouver la colonne (i.e., la variable) ayant le coût réduit le meilleur.

Le coût réduit C_{ks} va être défini de la façon suivante :

$$C_{ks} = P_{ks} - \mu_k - \sum_i \sum_j a_{ijks} \cdot \lambda_{ij}.$$

L'objectif du problème auxiliaire consiste donc à minimiser le coût réduit associé aux colonnes potentiellement entrantes. Les contraintes prises en compte pour ce problème sont toutes les contraintes intrinsèques à un horaire.

1.3.3 Particularités du problème maître

Dans certains cas, il peut arriver que les quarts de travail S_j ne forment pas une partition de la journée, mais se recoupent. Pour contourner ce problème, on introduit alors un nouvel objet appelé période de quota. L'ensemble des périodes est défini de façon à former une partition de la journée de travail de cardinalité minimale. Ce recouvrement est illustré par la figure 1.1. Sur celle-ci, les quarts de travail (D, D12, E1, ...) sont représentés par des lignes horizontales symbolisant leur durée). On constate qu'il y a recouvrement entre ces quarts de travail. On définit donc les périodes $P1$ à $P4$.

Ceci permet aux hôpitaux d'exprimer leurs contraintes de quotas non plus sur les quarts de travail qui ne forment plus une partition mais sur les périodes de quotas, qui sont créées de telle sorte qu'elles forment une partition de la journée.

Nous introduisons donc l'objet P_h, qui représente la h – ième période de

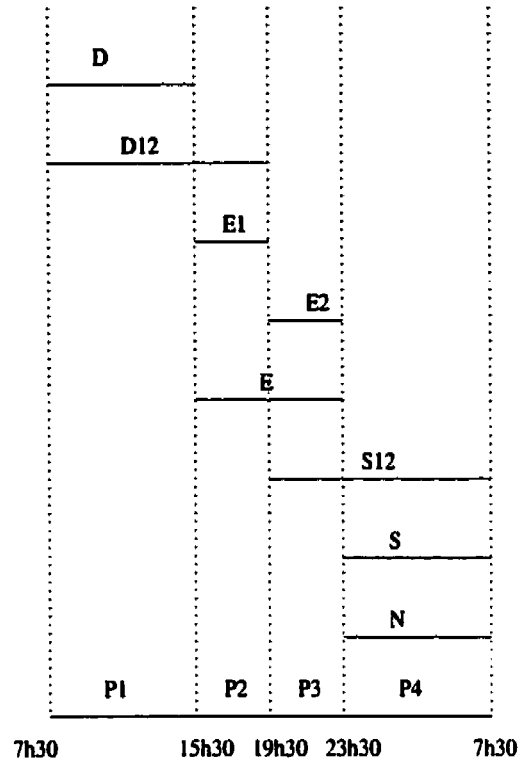


Figure 1.1: Lien quarts de travail - périodes de quotas

quotas de la journée. On note H l'ensemble des valeurs possibles pour l'indice h. On introduit la variable p_{hj} définie de la manière suivante:

$$p_{hj} = \begin{cases} 1 & \text{si la période } P_h \text{ est couverte par le quart } S_j, \\ 0 & \text{sinon.} \end{cases}$$

Les contraintes de quotas sont maintenant exprimées en termes de périodes. Il faut donc modifier certains objets de modélisation.

- La valeur cible de la contrainte de quota sera notée Q_{ih}.
- Les déficits et surplus maximaux seront respectivement notés $Q_{ih}^{max\_deficit}$ et $Q_{ih}^{max\_surplus}$.
- La variable d'écart sur chaque contrainte de quota sera notée s_{ih}.

Les contraintes sont modifiées de la façon suivante:

- Le respect des contraintes de quotas :

$$\forall i \in I, \forall h \in H, \quad \sum_k \sum_s \sum_j a_{ijks} \cdot p_{hj} \cdot y_{ks} - s_{ih} = Q_{ih}.$$

- Le respect de la fenêtre maximale de quota :

$$\forall i \in I, \forall h \in H, \quad -Q_{ih}^{max\_deficit} \leq s_{ih} \leq Q_{ih}^{max\_surplus}.$$

Pour résumer, le problème maître avec périodes de quota se formulera de la manière suivante:

$$\text{min.} \sum_k \sum_s P_{ks} \cdot y_{ks}$$

$$\text{s.l.c.} \begin{cases} \sum_k \sum_s \sum_j a_{ijks} \cdot p_{hj} \cdot y_{ks} - s_{ih} = Q_{ih} & i \in I, h \in H \\ \sum_s y_{ks} = 1 & k \in K \\ -Q_{ih}^{max\_deficit} \leq s_{ih} \leq Q_{ih}^{max\_surplus} & i \in I, h \in H \\ y_{ks} \in \{0,1\} & k \in K, s \in S. \end{cases}$$

Pour des raisons de simplification et parce que les deux modèles sont semblables, on travaillera sauf mention contraire avec le modèle dans lequel les contraintes de quotas sont exprimées en termes de quarts de travail.

1.3.4 Problème auxiliaire

Dans cette sous-section, nous allons présenter au lecteur la modélisation retenue pour résoudre le problème auxiliaire. Ce problème consiste à générer, pour une infirmière N_k sélectionnée par le problème maître, un ou plusieurs horaires réalisables au sens des contraintes intrinsèques à cette infirmière. Nous allons donc commencer par une description des données dont nous disposons pour les infirmières, ce qui nous permettra de définir les contraintes intervenant dans le problème auxiliaire. Nous définirons ensuite la modélisation retenue pour la résolution de ce problème.

Dans la suite de cette section, nous nommerons *affectation* un couple (jour D_i, quart de travail S_j) correspondant à l'affectation dans l'horaire courant du

quart de travail S_j le jour D_i pour l'infirmière considérée. Nous noterons cet objet A_{ij}.

1.3.4.1 Contraintes intrinsèques à une infirmière

Mis à part des renseignements permettant aux hôpitaux d'identifier leur personnel, nous disposons de données correspondant à des contraintes pour la résolution du problème auxiliaire. Ces contraintes peuvent être réparties en deux catégories :

- les contraintes dures, que l'on doit absolument satisfaire pour obtenir un horaire réalisable;
- les contraintes dites molles, qui n'interviendront que sous forme de pénalités introduites dans le coût d'un horaire en cas de violation de ces contraintes par l'horaire considéré.

Nous allons donc, pour chaque catégorie, décrire les différentes contraintes intervenant pour le problème considéré.

- Les contraintes dures, au nombre de sept, contrôlent la réalisabilité d'un horaire donné. Ces contraintes sont :
  - la liste des quarts de travail S_j que peut effectuer l'infirmière N_k;
  - une liste (pouvant être vide) d'affectations A_{ij} obligatoires ou interdites qui doivent être prises en compte dans tous les horaires potentiels générés pour l'infirmière N_k;
  - la règle régissant l'affectation des jours non travaillés et non payés (ceci ne concerne pas les jours de congés). Cette contrainte, exprimée à l'aide de deux entiers, précise quelle doit être la longueur (minimum et maximum) d'une série de jours consécutifs non travaillés non payés;
  - la charge de travail que doit recevoir l'infirmière sur des sous-périodes données. Cette charge sera exprimée en heures, chaque quart de travail S_j étant caractérisé par une durée exprimée en heures. Pour exprimer

cette contrainte, on divise l'horizon de planification en sous-périodes formant une partition de l'horizon, et on exprime pour chaque sous-période une borne inférieure et une borne supérieure sur la charge de travail que doit recevoir l'infirmière pendant cette sous-période. Au choix des hôpitaux, le temps de pause sera ou non compris dans cette charge de travail;

- les règles d'enchaînement des fins de semaines travaillées ou non. Cette règle permet de définir quelles sont les rotations entre fins de semaine travaillées / non travaillées que l'infirmière doit effectuer. Cette contrainte est exprimée par l'intermédiaire de quatre entiers : une borne inférieure et une borne supérieure sur le nombre de fins de semaine consécutives travaillées, et une borne inférieure et une borne supérieure sur le nombre de fins de semaine consécutives non travaillées. Par exemple, pour modéliser le fait de travailler une fin de semaine sur deux, on fixera les quatre bornes à la valeur unitaire. Pour modéliser le fait de travailler une fin de semaine sur trois, les deux bornes (min et max) sur le nombre de fins de semaine travaillées seront égales à 1, les deux bornes sur les fins de semaine de congés étant égales à 2;
- les règles d'enchaînement des différents types de quart. En effet, chaque quart de travail est caractérisé par un type (jour, soir ou nuit), et des contraintes doivent être respectées pour l'enchaînement de ces différents types de quarts dans l'horaire à produire pour l'infirmière considérée. Cette contrainte est exprimée sous forme de six entiers représentant les bornes inférieures et supérieures sur le nombre de quarts successifs de même type que l'on peut affecter à cette infirmière;
- la règle contrôlant l'affectation de jours de congés payés. Cette contrainte spécifie les nombres minimaux et maximaux de jours de congés payés à accorder, ainsi que la liste des jours candidats pouvant être

accordés en tant que congés.

- Les contraintes dites souples n'interviennent que sous forme de pénalités associées le cas échéant au coût de l'horaire généré. Ces contraintes, au nombre de deux, sont les suivantes :
  - la contrainte contrôlant les ratios entre les différents types de quart. En effet, il est préférable que l'horaire pris dans sa globalité soit relativement balancé en terme de types de quarts (jour, soir ou nuit). Pour ce faire, six entiers indiquant des bornes inférieures et supérieures pour les trois types de quarts de travail sont exprimés. Si l'horaire généré ne respecte pas ces ratios, une pénalité est introduite dans le coût de l'horaire;
  - les préférences et aversions exprimées par l'infirmière. En effet, celle-ci peut exprimer des préférences (positives ou négatives) quant aux affectations qui lui seront attribuées pour l'horizon courant. Tant que faire se peut, le programme essaiera de respecter ces préférences exprimées. En cas de non-respect de ces contraintes, une pénalité est introduite dans le coût de l'horaire généré.

1.3.4.2 Modélisation retenue pour le problème auxiliaire

Pour résoudre le problème auxiliaire, qui consiste à générer un ou des horaires réalisables pour une infirmière sélectionnée par le problème maître, nous faisons appel à une approche de type graphe. Plus précisément, nous utiliserons un algorithme de plus courts chemins dans un graphe acyclique avec fenêtres de ressources pour résoudre notre problème. Ce type d'algorithme est devenu classique en recherche opérationnelle et a déjà été étudié de manière intensive dans la littérature (voir [12] et [1]).

Il s'agit de construire un graphe $G = (V, E)$ dont le parcours est contrôlé

par des vecteurs de ressource. Les noeuds servent à la mise à jour des valeurs de ressources, et les arcs servent à autoriser ou refuser un passage suivant la valeur courante de la ressource. Après un algorithme servant à construire un graphe $G\prime$ des états réalisables [2], on applique un classique algorithme de plus courts chemins. La définition précise du graphe est donnée à la section 2.2.

L'adaptation de ce type d'algorithme à notre problème est décrite dans le chapitre concernant la gestion du problème auxiliaire (voir chapitre 2).

1.4 Résolution du problème

Le problème maître qui nous est donné à résoudre est un problème NP-difficile. Il est rendu complexe par le nombre très élevé des variables de décision y_{ks} qui interviennent. On rappelle au lecteur la définition de ces variables :

$$y_{ks} = \begin{cases} 1 & \text{si l'horaire } H_{ks} \text{ est conservé dans la solution optimale} \\ 0 & \text{sinon} \end{cases}$$

Le problème maître, rappelons-le, est formulé comme suit :

$$\text{min.} \sum_k \sum_s P_{ks} \cdot y_{ks}$$

$$\text{s.l.c.} \begin{cases} \sum_k \sum_s a_{ijks} \cdot y_{ks} - s_{ij} = Q_{ij} & i \in I, j \in J \\ \sum_s y_{ks} = 1 & k \in K \\ -Q_{ij}^{max\_deficit} \leq s_{ij} \leq Q_{ij}^{max\_surplus} & i \in I, j \in J \\ y_{ks} \in \{0,1\} & k \in K, s \in S. \end{cases}$$

Il s'agit donc d'un problème en nombres entiers. Pour le résoudre, on va faire appel à deux algorithmes très connus en recherche opérationnelle. Nous utiliserons donc dans un premier temps un algorithme dit de séparation et évaluation progressive ([10], [11]), ce qui permettra de n'avoir à résoudre que des versions relaxées du problème de départ. Cependant, pour résoudre ces problèmes relaxés, le nombre de variables étant encore très élevé, on fera appel à un deuxième algo-

rithme classique de recherche opérationnelle : la méthode dite de génération de colonnes.

Nous allons donc, dans le reste de cette section, détailler plus avant ces deux algorithmes ou schémas de résolution utilisés.

1.4.1 Algorithme de séparation et évaluation progressive

Comme on le sait, il est plus facile de résoudre un problème dans lequel les contraintes d'intégralité ont été relaxées qu'un problème en nombres entiers. C'est la motivation principale qui réside derrière l'adoption de la méthode de séparation et évaluation progressive.

Dans notre cas, on va relaxer la variable décisionnelle du problème maître. Le problème devient alors :

$$\text{min.} \sum_k \sum_s P_{ks} \cdot y_{ks}$$

$$\text{s.l.c.} \begin{cases} \sum_k \sum_s a_{ijks} \cdot y_{ks} - s_{ij} = Q_{ij} & i \in I, j \in J \\ \sum_s y_{ks} = 1 & k \in K \\ -Q_{ij}^{max\_deficit} \leq s_{ij} \leq Q_{ij}^{max\_surplus} & i \in I, j \in J \\ y_{ks} \in [0,1] & k \in K, s \in S. \end{cases}$$

La résolution de ce problème va nous donner une borne inférieure sur la valeur de la solution optimale du problème de départ en nombres entiers. En effet, ayant à résoudre une version moins contrainte du problème, l'espace de recherche est plus grand et la solution optimale du problème relaxé est meilleure ou égale à celle du programme en nombres entiers. Cette borne inférieure nous permettra la plupart du temps de couper des branches dans l'arbre d'exploration résultant des différentes contraintes de branchement que nous décrirons plus loin dans cette section. Si, de plus, par chance, l'optimum de la version relaxée de notre problème est atteint pour une solution entière, alors on peut en déduire que l'on a atteint l'optimum pour le problème en nombres entiers considéré au départ.

Pour les problèmes utilisant la génération de colonnes (voir section 1.4.2), il est connu qu'il est plus efficace de brancher sur les variables du problème auxiliaire plutôt que sur celles du problème maître. C'est pourquoi nous avons retenu le schéma de branchement suivant. Pour s'assurer de l'existence d'un branchement en un noeud donné, nous utilisons deux stratégies de branchement.

- La première est dite de type Ryan-Foster [13]. Le branchement revient alors à choisir une infirmière N_k, pour laquelle on dispose de deux horaires distincts H_{ks_1} et H_{ks_2} tels que $y_{ks_1} \approx 0.5$ et $y_{ks_2} \approx 0.5$. On cherche alors deux affectations $A_{ij} \in H_{ks_1}$ et $A_{i'j'} \in H_{ks_2}$ telles que :

$$a_{ijks_1} = a_{ijks_2}$$

et

$$a_{i'j'ks_1} \neq a_{i'j'ks_2}$$

Le branchement est alors défini par :

  - branche 1 : on force à ce que les deux affectations soient présentes ou absentes simultanément dans tous les horaires générés pour cette infirmière. Soit :

$$a_{ijks} = a_{i'j'ks}$$

  - branche 2 : on force à ce qu'une et une seule des affectations soit présente. Soit :

$$a_{ijks} \neq a_{i'j'ks}$$

- Nous verrons ci-après (voir proposition 1.4.1) qu'il n'existe pas forcément de branchement de type Ryan-Foster possible. Un deuxième type de branchement, choisi si on ne trouve pas de branchement Ryan-Foster, est un branchement binaire simple. On choisit alors une infirmière N_k, et deux horaires H_{ks_1} et H_{ks_2} tels que $y_{ks_1} \approx 0.5$ et $y_{ks_2} \approx 0.5$. On choisit alors une affectation

A_{ij} telle que :

$$a_{ijks_1} \neq a_{ijks_2}$$

Le branchement est alors défini par :

- branche 1 : on force à ce que l'affectation soit présente dans tous les horaires générés pour l'infirmière N_k :

$$\forall s, \quad a_{ijks} = 1$$

- branche 2 : on force à ce que l'affectation soit interdite dans tous les horaires générés pour l'infirmière N_k :

$$\forall s, \quad a_{ijks} = 0$$

Proposition 1.4.1 Dans le branchement de type Ryan-Foster, on peut toujours trouver une affectation $A_{i'j'}$ satisfaisant les contraintes pour la deuxième condition. Par contre, on n'est pas assuré de pouvoir trouver une affectation A_{ij} satisfaisant les contraintes de la première condition.

Preuve Commençons par prouver le première partie de la proposition. Soient H_{ks_1} et H_{ks_2} deux horaires distincts pour l'infirmière N_k tels que $y_{ks_1} \approx 0.5$ et $y_{ks_2} \approx 0.5$. La proposition nous dit que :

$$\exists (i', j') \in I \times J \text{ tel que } a_{i'j'ks_1} \neq a_{i'j'ks_2}.$$

La preuve de cette affirmation se fait en raisonnant par l'absurde. Le contraire logique de l'énoncé ci-dessus s'écrit :

$$\forall (i', j') \in I \times J, a_{i'j'ks_1} = a_{i'j'ks_2}.$$

Ce qui signifie que les deux horaires sont identiques en tous points. Ceci contredit l'hypothèse selon laquelle les horaires sont distincts. La première partie de la proposition est donc démontrée.

La deuxième partie de la proposition se montre facilement en exhibant un contre exemple : pour qu'il n'existe pas d'affectation A_{ij} dans les horaires H_{ks_1} et H_{ks_2} telle que $a_{ijks_1} = a_{ijks_2}$, il suffit qu'il n'y ait qu'un seul quart de travail réalisable pour l'infirmière N_k considérée, et que le deuxième horaire soit exactement le négatif du premier (en termes quart de travail et jour de repos). Ce cas est certes très rare, mais il peut exister et nous oblige donc à avoir recours à un deuxième schéma de branchement pour assurer la couverture correcte de tous les cas possibles de notre algorithme de séparation et évaluation progressive.

1.4.2 Algorithme de génération de colonnes

Du fait du choix de variables que nous avons fait pour ce problème, leur nombre est exponentiel en fonction du nombre de jours présents dans l'horizon à considérer et du nombre de quarts de travail réalisables. Dans ce cas de figure, il est impensable de générer toutes les variables du problème (i.e., tous les horaires réalisables pour toutes les infirmières) à priori. On fait donc appel à un algorithme classique de recherche opérationnelle, la génération de colonnes (Nemhauser et Wolsey [10], Chvátal [14]).

Cet algorithme est basé sur la méthode révisée du simplexe, et va faire appel à une décomposition du problème de départ en deux sous-problèmes : un problème maître (comme vu plus haut) et un problème auxiliaire. Le rôle du problème auxiliaire sera donc de fournir à chaque itération du simplexe une variable entrante (ou colonne entrante) au problème maître. Le but du problème auxiliaire va donc être de générer, pour une infirmière sélectionnée par le problème maître, un ou des horaires réalisables. La liaison entre les deux problèmes sera effectuée par les variables duales associées aux différentes contraintes du problème maître. Celles-ci permettront de générer à chaque étape des horaires qui répondront le mieux possible à la satisfaction courante des contraintes de quotas et de partitionnement.

1.4.3 Cas de non réalisabilité

Dans le cas où les données proposées conduisent à un problème n'admettant aucune solution, on désire que le logiciel développé (IRIS) soit suffisamment polyvalent pour proposer à l'utilisateur certaines solutions.

Plusieurs cas de non réalisabilité peuvent se présenter.

- Les contraintes intrinsèques d'une infirmière N_k conduisent à une non réalisabilité au niveau de l'horaire à générer pour cette infirmière. Dans ce cas, le programme affichera l'infirmière concernée, ne lui génèrera pas d'horaire mais continuera l'optimisation avec le reste des infirmières de l'équipe.
- Le problème est sur-contraint au niveau des quotas (trop peu d'infirmières pour remplir tous les quotas). Dans ce cas le programme ne génèrera pas d'horaires, mais l'utilisateur peut demander la sortie de la solution qui, jusqu'alors, viole le moins les contraintes de quotas.
- Le problème est sous-contraint au niveau des quotas (trop d'infirmières présentes par rapport aux quotas exprimés). Dans ce cas, le programme sortira des horaires dans lesquels certaines infirmières n'ont pas d'horaire de travail pour l'horizon considéré.

Chapitre 2

Problème auxiliaire : description du graphe

2.1 Introduction

Comme nous l'avons vu plus tôt, le problème auxiliaire est constitué d'un problème de plus courts chemins dans un graphe acyclique avec présence de contraintes de ressources. Ce type de problème a déjà été étudié et plusieurs solutions sont disponibles pour le résoudre. Nous utiliserons une méthode classique dont l'idée avait été développée par Jaumard, Semet et Vovor ([1], [2]).

Cette technique est basée sur une formulation des diverses contraintes dures en ressources qui contrôlent les chemins que l'on peut emprunter lors du parcours du graphe. Ce graphe étant destiné à générer un horaire réalisable pour une infirmière, il existe un graphe différent par infirmière considérée dans le problème maître.

Dans un premier temps, nous étudierons comment, à partir d'une sélection de contraintes de base, nous construisons un graphe dont les sommets représentent les diverses affectations possibles pour une infirmière donnée. Nous détaillerons ensuite les règles qui gouvernent la construction des arcs du graphe.

Nous décrirons ensuite les quatre ressources qui contrôlent le parcours du graphe. Nous montrerons comment ces ressources sont modélisées à partir des contraintes dures intervenant dans le problème auxiliaire. L'accent sera mis sur l'étude des différents cas pouvant se présenter, nous justifierons ainsi la complétude de cette étude.

2.2 Construction du graphe

Comme nous l'avons vu plus haut, un graphe est construit pour chaque infirmière présente dans le problème maître (chaque infirmière pour qui nous avons un horaire à générer). Soit N_k cette infirmière. Les règles de base présidant à la construction du graphe sont décrites ci dessous.

- Un sommet v_ℓ représente, pour l'infirmière considérée, une affectation potentielle A_{ij} d'un quart de travail pour une journée précise (jour D_i, quart S_j).
- Un arc (v_{ℓ_1}, v_{ℓ_2}) sera présent dans le graphe si l'infirmière peut réaliser l'affectation $A_{i_2 j_2}$ après avoir réalisé l'affectation $A_{i_1 j_1}$. Là encore, ce sont les contraintes dures (voir section 2.2.2) qui contrôlent la construction des arcs.

Nous allons dans les deux prochaines sections détailler les règles qui gouvernent en détail à la construction des noeuds et des arcs présents dans le graphe.

2.2.1 Définition des noeuds

Comme vu plus haut, un noeud v_ℓ représente une affectation possible pour l'infirmière considérée, c'est-à-dire un objet affectation noté A_{ij} défini comme une paire (jour D_i, quart de travail S_j). Certaines règles président à l'existence d'un noeud donné. Ces règles sont les suivantes :

- le jour D_i doit appartenir à l'horizon de planification;

- le quart de travail S_j doit appartenir à la liste des quarts réalisables pour l'infirmière considérée;
- l'affectation A_{ij} ne doit pas appartenir à la liste des affectation interdites pour l'infirmière considérée.

Si ces trois règles sont satisfaites, alors le noeud v_ℓ qui représente l'affectation A_{ij} appartiendra au graphe correspondant au problème auxiliaire pour l'infirmière N_k considérée.

Pour le noeud source du graphe, il existe deux possibilités : si on dispose de l'horaire réalisé pendant la période précédant l'horizon de planification courant, la source représente la dernière affectation de cet horizon. Sinon, la source est un noeud portant un quart de travail et un jour virtuels.

De même, si on dispose de l'horizon précédent, les valeurs initiales des ressources seront héritées des valeurs finales de l'horizon précédent. Sinon, les valeurs initiales seront égales à 0.

2.2.2 Construction des arcs

Comme vu plus haut, un arc existe dans le graphe entre deux sommets v_{ℓ_1} (affectation $A_{i_1 j_1}$) et v_{ℓ_2} (affectation $A_{i_2 j_2}$) si l'enchaînement des affectations représentées par les deux sommets est possible par rapport aux contraintes dures de l'infirmière considérée. Les règles utilisées pour décider de l'existence d'un arc (v_{ℓ_1}, v_{ℓ_2}) sont décrites ci-dessous.

- Le jour D_{i_1} doit être différent du jour D_{i_2}. En effet, la convention collective des infirmières interdit de recevoir deux affectations le même jour.
- Les règles d'enchaînement des quarts de travail doivent être respectées :
  - si D_{i_2} est le lendemain de D_{i_1}, alors S_{j_2} doit appartenir à la liste des quarts de travail réalisables le lendemain du quart S_{j_1};
  - si D_{i_2} n'est pas le lendemain de D_{i_1}, alors S_{j_2} doit appartenir à la liste

des quarts de travail réalisables après avoir reçu des jours non travaillés après un quart S_{j_1}.

- Si D_{i_2} n'est pas le lendemain de D_{i_1}, alors le nombre de jours situés entre D_{i_1} et D_{i_2} doit être compris entre le nombre minimum de jours non travaillés consécutifs et le nombre maximum de jours non travaillés consécutifs pour cette infirmière. Le cas particulier des congés payés à accorder est traité en section 2.3.2.2.
- S'il y a des fins de semaine complètes (samedi et dimanche) non travaillées entre D_{i_1} et D_{i_2}, le nombre de ces fins de semaine doit être compris entre le nombre minimum et le nombre maximum de fins de semaine consécutives non travaillées que l'infirmière peut recevoir.
- S'il y a des sous-périodes de charge de travail (voir section 2.3 pour la définition) entières évitées par l'arc (v_{ℓ_1}, v_{ℓ_2}) , alors la charge de travail minimale pour ces sous-périodes doit être égale à zéro.

Si ces cinq règles sont satisfaites, alors il existe un arc entre les sommets v_{ℓ_1} et v_{ℓ_2}.

Notes :

- Si on ne dispose d'aucune information concernant l'horizon précédent, on crée un noeud source virtuel dans le graphe. Dans ce cas, les règles faisant intervenir le jour D_{i_1} sont modifiées. À la place du jour D_{i_1}, on substitue dans ces règles la veille du premier jour de l'horizon courant.
- Pour le puits, les règles faisant intervenir le jour D_{i_2} sont modifiées. À la place du jour D_{i_2}, on utilise le lendemain du dernier jour de l'horizon courant pour vérifier la valeur des différentes règles.

2.2.3 Données présentes sur les noeuds et les arcs du graphe

Dans cette section, nous présentons au lecteur les différentes données présentes sur les noeuds et les arcs du graphe, et qui servent à contrôler le passage sur les différents arcs du graphe [2].

Sur les noeuds, on dispose de deux intervalles pour chaque ressource notés $X_r = [\underline{X}_r, \overline{X}_r]$ et $H_r = [\underline{H}_r, \overline{H}_r]$, r désignant l'indice de la $r-$ième ressource. Ces intervalles servent à la mise à jour de la valeur courante de la ressource considérée. La fonction de mise à jour s'écrit de la façon suivante (on note x_r la valeur courante de la ressource R_r avant l'arrivée sur le noeud considéré et x_r^{new} la valeur courante de la ressource après mise à jour sur le noeud considéré):

$$x_r^{new} = \begin{cases} \underline{X}_r & si \quad x_r < \underline{H}_r \\ x_r & si \quad \underline{H}_r \leq x_r \leq \overline{H}_r \\ \overline{X}_r & si \quad \overline{H}_r \leq x_r. \end{cases}$$

Sur les arcs, on trouve pour chaque ressource un intervalle $\omega_r = [\underline{\omega}_r, \overline{w}_r]$ et un entier u_r. Le contrôle de passage sur l'arc s'effectue de la façon suivante (x_r^{new} désigne ici la valeur courante de la ressource à la sortie du noeud origine, et $x_r{}'$ la valeur courante à l'entrée dans le noeud destination):

- si $x_r^{new} < \underline{\omega}_r$ ou $x_r^{new} > \overline{\omega}_r$, le passage sur l'arc est refusé;
- si $\underline{\omega}_r \leq x_r^{new} \leq \overline{\omega}_r$, le passage est autorisé sur l'arc et on a $x_r{}' = x_r^{new} + u_r$.

Précisons que la valeur de u_r se rapporte toujours au noeud destination de l'arc (par exemple, pour la ressource $R1$, u_1 représentera sur chaque arc la durée du quart porté par le noeud destination).

2.2.4 Comparaison avec d'autres méthodes

L'algorithme de plus courts chemins avec contraintes de ressources a déjà été étudié de très nombreuses fois dans la littérature ([2], [1] et [12]). De nombreuses méthodes sont apparues pour formuler et résoudre ce type de problèmes.

Desaulniers *et al.* ont proposé [12] un algorithme basé sur une formulation différente de celle employée ici. La méthode décrite présente un cadre général pour résoudre les problèmes se formulant comme des problèmes de plus courts chemins avec contraintes de ressources. Cette unification est une extension de la méthode présentée par Desrochers [15].

Cette méthode a abouti sur le développement puis la commercialisation d'un outil nommé Gencol. Dans cette approche, les contraintes de parcours ne sont pas placées sur les arcs mais sur les noeuds. De plus, le contrôle de la valeur courante de la ressource ne se fait que grâce à une borne supérieure. Par contre, cette perte de généralité est compensée par le fait que la fonction de mise à jour peut être toute fonction non décroissante, et n'est plus limitée à une fonction affine.

Les deux modèles sont très proches, mais pas identiques. Notre problème peut être résolu grâce à Gencol, moyennant quelques précautions. Nous allons étudier ci-dessous comment notre problème pourrait être formulé pour être résolu grâce à Gencol.

- Avec Gencol, le contôle de valeur s'effectue sur les noeuds du graphe. Une solution pour remédier à ce problème serait de créer sur tous les arcs un noeud fictif qui contiendrait la borne supérieure présente auparavant sur l'arc.
- Gencol ne considérant que les bornes supérieures pour les contraintes de parcours, il serait nécessaire de dédoubler toutes les ressources dont les bornes inférieures ne sont pas triviales (i.e. différentes d'une valeur constante sur tous les arcs). On aurait donc un dédoublement du nombre de ressources

dans le graphe, donc un alourdissement considérable du problème.

- Gencol ne prenant pas en compte les mises à jour de valeur de la ressource autrement que par la fonction de consommation, il faudrait modifier les consommations pour leur faire intégrer les fenêtres de mises à jour présentes auparavant sur les noeuds. Cependant, c'est ici que se présente la première incompatibilité entre les deux modèles : on rappelle que la fonction de consommation doit être croissante. Considérons l'exemple suivant : sur l'arc d'arrivée, $u = 2$, sur le noeud considéré, $H = [3,6]$ et $X = [4,5]$. On note x la valeur de la ressource avant passage sur l'arc puis le noeud, et $f(x)$ la valeur en sortant du noeud considéré. On a alors les résultats suivants :

  - si $x < 1$, $f(x) = 4$,
  - si $1 \leq x \leq 4$, $f(x) = x + 2$,
  - si $5 \leq x$, $f(x) = 5$.

  Comme on peut le constater, la fonction f n'est pas monotone, c'est-à-dire ni croissante ni décroissante. Il n'est donc pas possible d'appliquer le modèle développé pour Gencol. Cependant, dans le cas particulier où, sur tous les noeuds, on a $H = X$, alors on pourra appliquer la méthode Gencol.

Proposition 2.2.1 Si, sur tous les noeuds du graphe considéré, on a $H = X$, alors on peut formuler une fonction de consommation (au sens de Gencol) non décroissante et ainsi appliquer la méthode sus-citée à notre problème, moyennant quelques reformulations mineures.

Preuve Si, sur un noeud courant, on a $H = X = [\underline{X}, \overline{X}]$, on doit avoir une fonction de consommation donnant les résultats suivants :

$$f(x) = \begin{cases} \underline{X} & si \quad x + u < \underline{X} \\ x + u & si \quad \underline{X} \leq x + u \leq \overline{X} \\ \overline{X} & si \quad \overline{X} \leq x. \end{cases}$$

Soit la fonction g définie comme suit :

$$g(x) = \underline{X} + max(0, min(\overline{X} - \underline{X}, x + u - \underline{X}))$$

On a alors la discussion suivante :

$$g(x) = \begin{cases} \underline{X} & si \quad x + u < \underline{X} \\ x + u & si \quad \underline{X} \leq x + u \leq \overline{X} \\ \overline{X} & si \quad \overline{X} \leq x. \end{cases}$$

On a donc bien $\forall x \in \mathcal{N}, f(x) = g(x)$. De plus, la fonction g est monotone croissante. Il existe donc bien une fonction de consommation (au sens de Gencol) monotone. On peut donc, moyennant reformulation du problème, appliquer l'algorithme Gencol, à condition bien sur d'avoir pour tous les noeuds du graphe $H = X$. Cependant, dans la pratique, la multiplication par 2 du nombre de ressources à utiliser freinera grandement la résolution de notre problème par un algorithme de type Gencol.

2.3 Ressource associée à la charge de travail

2.3.1 Définitions et hypothèses préalables

La première ressource que nous allons étudier est la ressource associée aux contraintes de charge de travail. Ces contraintes se formulent de la manière suivante : l'horizon de planification est divisé en une partition de sous-périodes $\mathcal{P}_m$. L'indice m représentera donc dans cette section le numéro de la sous-période considérée. Pour chaque sous-période $\mathcal{P}_m$, l'infirmière doit effectuer un nombre d'heures compris entre $\overline{R1}_m$ et $\underline{R1}_m$. La première ressource va nous permettre de contrôler le respect de ces contraintes.

Nous nommerons donc $R1$ cette ressource et la valeur courante du compteur associé à celle-ci sera notée x_1. Ce compteur modélise le nombre d'heures tra-

vaillées par l'infirmière dans la sous-période courante. Le but de cette ressource est de contrôler qu'à chaque changement de sous-période, l'infirmière a bien réalisé une charge de travail comprise entre sa charge minimale et sa charge maximale. Nous allons donc maintenant détailler le comportement de cette ressource dans notre graphe.

On formule l'hypothèse (justifiée par les expériences que nous avons eues avec les hôpitaux) selon laquelle la première sous-période commence le premier jour de l'horizon de planification. Ceci nous permet de postuler que sur le noeud source, on doit avoir $x_1 = 0$.

2.3.2 Étude des différents cas

Nous allons dans cette section étudier les différents types de cas auxquels nous pouvons être confrontés lors de l'exploration du graphe. Nous commencerons dans un premier temps par décrire et justifier les étiquettes portées par les noeuds du graphe, puis nous étudierons les arcs du graphe, en commençant par les cas généraux (c'est-à-dire ne faisant pas intervenir de noeuds particuliers). Nous examinerons ensuite les cas plus spécifiques (source, puits, noeuds fictifs,...). Une dernière partie de cette étude de cas sera consacrée au cas particulier où on doit affecter des jours de congés sur l'horizon de planification.

Durant cette étude de cas, nous utiliserons les notations décrites ici.

- les jours considérés seront notés D_1, D_2, ...
- les quarts de travail seront notés S_1, S_2, ...
- on notera $\ell(S_j)$ la durée (en unités de temps appropriée) du quart de travail S_j,
- pour chaque sous-période $\mathcal{P}_m$ de l'horizon, $\underline{Rl}_m$ et $\overline{Rl}_m$ représenteront les bornes inférieure et supérieure sur la charge de travail que doit effectuer l'infirmière considérée pour cette sous-période.

2.3.2.1 Étiquettes sur les noeuds

Pour cette première ressource, les étiquettes sur les noeuds sont assez simples et faciles à modéliser. Nous allons d'abord étudier le cas classique, puis nous verrons comment modéliser les fenêtres de ressources sur les noeuds particuliers du graphe.

Cas du noeud classique : jour et quart de travail non fictifs

Les règles suivantes s'appliquent :

- Pour la borne inférieure, la durée du quart S_j associé à ce noeud étant $\ell(S_j)$, on doit avoir $x_1 \geq \ell(S_j)$. On aura donc sur le noeud : $\underline{H} = \underline{X} = \ell(S_j)$.
- Pour la borne supérieure, le contrôle sur la valeur supérieure de x_1 s'effectuant sur les arcs, cette dernière n'est pas forcément utile. Mais pour des raisons d'adéquation au modèle on pose $\overline{X} = \overline{H} = \overline{R1}_m$ (on note que l'on aurait aussi bien pu mettre la valeur $+\infty$).

Cas de la source

On part de l'hypothèse précédente selon laquelle la valeur initiale de x_1 est 0. On affecte donc les intervalles suivants à la source : $H = X = [0, 0]$. La première sous-période de l'horizon débutant par hypothèse forcément le premier jour de l'horizon, avoir ou non des données concernant l'horizon précédent ne change rien. On doit toujours avoir $x_1 = 0$ sur la source.

Cas du puits

Là encore les étiquettes de ressource sur le noeud proprement dit sont inutiles. Pour réduire toutefois la complexité de l'algorithme de construction du plus courts chemins [2], on place les étiquettes suivantes sur le puits : $H = X = [0, \overline{R1}_{last}]$, $\overline{R1}_{last}$ représentant la limite supérieure sur la charge de travail que doit recevoir l'infirmière pour la dernière sous-période de l'horizon courant.

Cas des noeuds fictifs

Comme on le verra lors de la section suivante, les noeuds fictifs n'interviennent que pour contrôler certains cas particuliers liés aux fins de semaine. Ils ne doivent jouer aucun rôle pour les autres ressources. On pose alors les étiquettes suivantes sur un noeud fictif : $H = X = [0, +\infty]$.

Récapitulatif

- Noeud général appartenant à la sous-période $\mathcal{P}_m$: $H = X = [\ell(S_j), \overline{R1}_m]$.
- Source : $H = X = [0, 0]$.
- Puits : $H = X = [0, \overline{R1}_{last}]$.
- Noeud fictif : $H = X = [0, +\infty]$.

2.3.2.2 Étiquettes sur les arcs

Nous allons maintenant détailler les étiquettes intervenant sur les arcs, en commençant par les arcs généraux du graphe puis en poursuivant avec les arcs dits particuliers.

Cas 1a : les deux affectations appartiennent à la même sous-période $\mathcal{P}_m$

Ce cas est illustré par la figure 2.1.

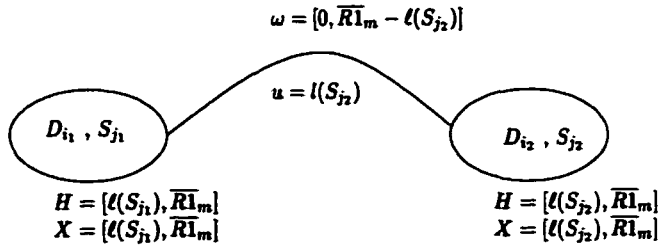


Figure 2.1: Cas de deux jours dans la même sous-période (ressource $R1$)

Dans ce cas, la consommation de ressource portée sur l'arc joignant les deux sommets sera $u = \ell(S_{j_2})$. Nous devons donc autoriser le passage sur cet arc si $0 \leq x_1 \leq \overline{R1}_m - \ell(S_{j_2})$. En effet, l'infirmière ne pourra effectuer l'affectation représentée par le noeud cible que si elle n'a pas déjà atteint son maximum d'heures pour la sous-période considérée. On rappelle que la mise à jour de la ressource s'effectue après le contrôle de passage. On aura donc sur l'arc les données suivantes:

$$\omega = [0, \overline{R1}_m - \ell(S_{j_2})]; u = \ell(S_{j_2}).$$

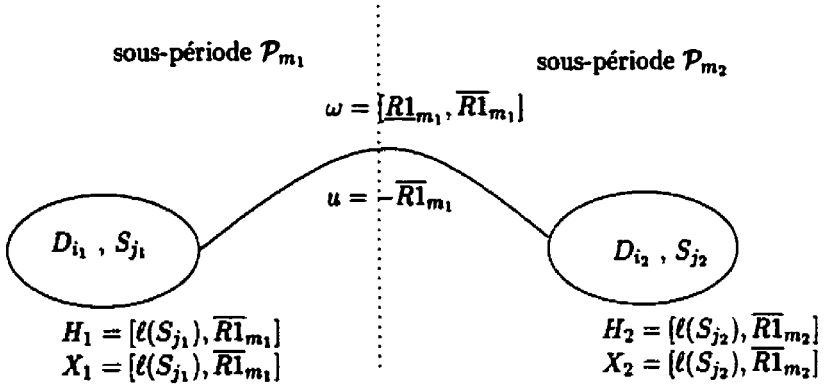


Figure 2.2: Cas de deux jours dans deux sous-périodes différentes (ressource $R1$)

Cas 1b : les deux affectations appartiennent à deux sous-périodes $\mathcal{P}_{m_1}$ et $\mathcal{P}_{m_2}$

Le second cas général considéré (figure 2.2) est le cas d'un arc représentant un changement de sous-période. Soit $\mathcal{P}_{m_1}$ la sous-période du jour de l'affectation représentée par le noeud origine, et $\mathcal{P}_{m_2}$ celle correspondant au noeud cible. Sur l'arc qui permet le changement de sous-période, il faut contrôler que l'infirmière a bien réalisé un nombre d'heures compris entre $\underline{R1}_{m_1}$ et $\overline{R1}_{m_1}$. Pour la consommation de ressource, il faut s'assurer de remettre à zéro la valeur courante de la ressource. On va donc poser $u = -\overline{R1}_{m_1}$, pour s'assurer d'arriver sur le noeud

cible avec $x_1 \leq 0$. Les étiquettes sur le noeud cible se chargeront de remettre x_1 à jour avec la durée du quart de travail représenté par le noeud cible. On porte donc les données suivantes sur l'arc :

$$\omega = [\underline{R1}_{m_1}, \overline{R1}_{m_1}]; u = -\overline{R1}_{m_1}.$$

Comme nous allons le constater, certains cas particuliers se posent lorsque le noeud origine de l'arc est la source, lorsque le noeud cible est le puits ou bien lorsqu'un noeud fictif se trouve sur l'arc. Nous allons dans cette section étudier en détails ces cas particuliers.

Cas 2a : le noeud origine est la source

Comme vu plus tôt lors de la définition du graphe, deux cas peuvent se présenter : la source peut représenter la dernière affectation de l'horaire précédent (si on dispose de celui-ci), ou bien la source peut être constituée d'un jour et d'un quart de travail fictifs (si on ne dispose pas de cet historique). Dans les deux cas, comme nous avons formulé l'hypothèse selon laquelle on commençait toujours le nouvel horaire avec $x_1 = 0$, le traitement de la ressource est le même. Nous étudierons donc le cas des arcs quittant la source dans un cadre générique.

Soit m l'indice de la sous-période à laquelle appartient le jour représenté par le noeud cible et m_{first} celui de la première sous-période de l'horizon courant. En fait, tout se passe comme si la source appartenait à la première sous-période de l'horizon. On retrouve donc la même discussion que pour les cas généraux, ce qui donne les résultats suivants :

- Si le jour du noeud cible appartient à la première sous-période de l'horizon courant (i.e., $m = m_{first}$), on a les étiquettes suivantes sur l'arc (S_{j_2} représente le quart porté par le noeud cible) :

$$\omega = [0, \overline{R1}_{m_{first}} - \ell(S_{j_2})]; u = \ell(S_{j_2}).$$

- Si le jour du noeud cible appartient à une période différente de la première période de l'horizon, l'infirmière prend une série de jours de congés couvrant une ou plusieurs périodes. Il n'y a pas de contrôle de validité de ressources à effectuer (on a $x_1 = 0$ à la source), car si l'arc existe, c'est que les contraintes dures correspondantes sont vérifiées (i.e. l'infirmière est autorisée à prendre ces congés). L'arc portera donc les mêmes étiquettes que précédemment (S_{j_2} représente le quart porté par le noeud cible) :

$$\omega = [0, \overline{R1}_m - \ell(S_{j_2})]; u = \ell(S_{j_2}).$$

Cas 2b : le noeud cible est le puits

Pour le noeud représentant le puits, la discussion est un peu la même que pour la source. Il s'agit de savoir si l'arc provient d'un noeud de la dernière sous-période de l'horizon ou bien d'une autre sous-période. On trouve donc les résultats décrits ci-bas.

- Si le jour du noeud origine appartient à la dernière sous-période m_{last} de l'horizon courant, il suffit de vérifier que l'infirmière a bien réalisé son quota de travail durant cette période. La consommation de ressource n'a aucune importance puisque le noeud cible est le puits, qui porte un jour et un quart de travail fictifs. On aura donc les données suivantes :

$$\omega = [\underline{R1}_{m_{last}}, \overline{R1}_{m_{last}}]; u = 0;$$

- Si le jour du noeud source appartient à une période d'indice m qui n'est pas la dernière de l'horizon, c'est que l'infirmière est autorisée à prendre toutes les sous-périodes subséquentes à $\mathcal{P}_m$ en congés. La phase de construction du graphe s'est chargée de vérifier par rapport aux contraintes une telle possibilité. Si l'arc existe, c'est que l'infirmière peut effectivement prendre de tels congés. Il faut donc vérifier sur l'arc que l'infirmière a bien rempli son quota pour la dernière sous-période travaillée. Là encore, la consommation

de ressource n'a pas d'importance puisque la cible de l'arc est le noeud puits. On trouve donc les données suivantes sur l'arc considéré :

$$\omega = [\underline{R1}_m, \overline{R1}_m]; u = 0.$$

Cas 2c : arc sur lequel est inséré un noeud fictif

Les noeuds fictifs sont utiles pour contrôler certaines particularités liées aux fins de semaines. Cependant, ils ne sont pas utiles pour les autres ressources. Soit donc v_{ℓ_1} le noeud origine, v_{ℓ_2} le noeud cible et v_{ℓ_f} le noeud virtuel inséré sur l'arc (v_{ℓ_1}, v_{ℓ_2}). Le noeud virtuel n'étant d'aucune utilité pour la première ressource, on choisit donc de faire figurer les informations suivantes :

- Sur l'arc (v_{ℓ_1}, v_{ℓ_f}), on porte (on suppose que le jour porté par v_{ℓ_1} appartient à la sous-période d'indice m) :

$$\omega = [0, \overline{R1}_m]; u = 0.$$

- Sur l'arc (v_{ℓ_f}, v_{ℓ_2}), on fait figurer les informations qui auraient dû se trouver sur l'arc (v_{ℓ_1}, v_{ℓ_2}) s'il n'y avait pas eu de noeud virtuel inséré (voir la définition des cas généraux pour la liste des différents cas pouvant se présenter).

Récapitulatif

- Les deux jours extrémités de l'arc appartiennent à la même sous-période $\mathcal{P}_m$:

$$\omega = [0, \overline{R1}_m - \ell(S_{j_2})]; u = \ell(S_{j_2}).$$

- Les deux jours extrémités de l'arc appartiennent à deux sous-périodes distinctes $\mathcal{P}_{m_1}$ et $\mathcal{P}_{m_2}$:

$$\omega = [\underline{R1}_{m_1}, \overline{R1}_{m_1}]; u = -\overline{R1}_{m_1}.$$

- Le noeud origine de l'arc est la source. Deux cas peuvent se présenter.
  - Le jour du noeud destination appartient à la première sous-période $\mathcal{P}_{m_{first}}$ de l'horizon :

$$\omega = [0, \overline{R1}_{m_{first}} - \ell(S_{j_2})]; u = \ell(S_{j_2}).$$

  - Le jour du noeud destination appartient à une sous période $\mathcal{P}_m$ différente de la première sous-période de l'horizon :

$$\omega = [0, \overline{R1}_m - \ell(S_{j_2})]; u = \ell(S_{j_2}).$$

- Le noeud cible de l'arc est le puits. Deux cas se présentent.
  - Le jour du noeud origine appartient à la dernière sous-période $\mathcal{P}_{m_{last}}$ de l'horizon :

$$\omega = [\underline{R1}_{m_{last}}, \overline{R1}_{m_{last}}]; u = 0.$$

  - Le jour du noeud origine appartient à une sous-période différente de la dernière sous-période de l'horizon :

$$\omega = [\underline{R1}_m, \overline{R1}_m]; u = 0.$$

- Un noeud virtuel v_{ℓ_f} a été ajouté sur un arc (v_{ℓ_1}, v_{ℓ_2}), et m est l'indice de la sous-période à laquelle appartient le jour porté par v_{ℓ_1} :

$$1^{\text{ère}} \text{ moitié} : \omega = [0, \overline{R1}_m]; u = 0;$$

$$2^{\text{ème}} \text{ moitié} : \text{étiquettes portées par } (v_{\ell_1}, v_{\ell_2}).$$

Cas particulier : l'infirmière doit prendre des congés durant la sous-période considérée

La ressource $R1$ correspondant aux nombres d'heures à effectuer étant une contrainte dure pour l'infirmière considérée, certains aménagements de modélisation

sont nécessaires pour pouvoir prendre en compte de manière correcte les jours de congés que l'infirmière peut recevoir. Ces jours de congés sont gérés par la ressource R4 (voir 2.6).

En fait, l'aménagement à faire est assez simple et tout à fait logique dans l'optique de jours de congés qui peuvent être pris à certaines dates. Si le jour potentiel pris en congé D_i appartient à la sous-période $\mathcal{P}_m$, il suffit de déduire la durée du quart de travail le plus long potentiellement réalisé par l'infirmière considérée du minimum d'heures travaillées pour la sous-période $\mathcal{P}_m$.

Ainsi, de la même manière, si on autorise l'infirmière à prendre toute une sous-période en congés, il suffit de mettre ce même minimum à zéro. De la sorte, avec les règles de construction du graphe vues plus haut, on pourra avoir un ou des arcs qui évitent tous les jours contenus dans la sous-période $\mathcal{P}_m$.

2.4 Ressource associée aux fins de semaine

La seconde ressource que nous allons étudier, dénotée $R2$ dans les algorithmes du programme, correspond à la gestion des fins de semaine travaillées ou non. En effet, dans chaque convention collective, il est spécifié que l'infirmière doit respecter un certain rythme de travail pour les fins de semaine. La plupart travaillent une fin de semaine sur deux, tandis que d'autres travaillent toutes les fins de semaine ou bien n'ont pas de préférences.

La ressource $R2$ va donc nous être utile pour gérer ces contraintes de fins de semaine. Dans un premier temps, nous allons voir certaines définitions et hypothèses qui vont nous permettre de mieux comprendre le problème à modéliser. Puis, comme dans la section précédente, nous verrons les différents cas logiques qui peuvent se présenter et comment ils ont été traités dans la construction du graphe et des vecteurs de ressources associés.

2.4.1 Définitions et hypothèses

Tout d'abord, notons que cette ressource peut être désactivée par un paramètre dans le fichier d'entrée [16].

Si cette ressource est désactivée, il n'y aura aucun contrôle lors du plus courts chemins sur les fins de semaine, c'est-à-dire que toutes les possibilités d'enchaînement fins de semaine travaillées ou accordées en congés sont admissibles pour l'infirmière considérée.

Si la ressource est activée, on adopte les notations suivantes :

- x_2 dénotera la valeur courante du pointeur associé à cette ressource lors du parcours du graphe;
- pour l'infimière considérée, on dispose des informations suivantes :
  - Les nombres minimum et maximum de fins de semaine consécutives travaillées (respectivement notés $\underline{R2_{on}}$ et $\overline{R2_{on}}$).
  - Les nombres minimum et maximum de fins de semaine consécutives non travaillées (respectivement notées $\underline{R2_{off}}$ et $\overline{R2_{off}}$).

Lors du parcours du graphe, les conventions suivantes seront employées :

- Pour chaque fin de semaine travaillée (samedi et/ou dimanche), on incrémentera la valeur de x_2 d'une unité.
- Pour chaque fin de semaine non travaillée (ni samedi ni dimanche), on décrémentera la valeur de x_2 d'une unité.
- A chaque changement de type d'enchaînement (passage d'un cycle fin de semaine travaillées à un cycle de fins de semaine congés ou inversement), on remet la valeur de x_2 à zéro. Ceci a pour effet de dégager la propriété suivante : *une valeur positive de x_2 indique que l'on se situe dans un cycle de fins de semaine travaillées. Une valeur négative signale une série de congés.*

Note : Dans la suite, nous appellerons jour de semaine un jour n'appartenant

pas à la fin de semaine (c'est-à-dire un jour entre le lundi et le vendredi inclusivement).

2.4.2 Étude des différents cas

Comme on va le voir dans le présent chapitre, cette contrainte est la plus compliquée des ressources à gérer dans notre problème, donnant lieu parfois à la duplication d'arcs dans le graphe ou même à la création de noeuds fictifs comme on le verra plus loin. Dans un premier temps, nous allons voir comment les étiquettes sur les noeuds sont définies.

Nous étudierons ensuite les arcs et les étiquettes qu'ils doivent porter pour forcer le respect des différentes contraintes. Nous commencerons par les cas généraux qui peuvent se présenter, puis nous verrons ensuite les cas particuliers (arcs partant de la source, arrivant au puits, arcs dédoublés,...). Des schémas seront fournis pour permettre au lecteur de bien comprendre les différents cas étudiés et se convaincre de la complétude de cette étude.

2.4.2.1 Étiquettes portées par les noeuds

De manière générale par rapport à la ressource considérée ici, deux cas peuvent se présenter pour les noeuds recontrés sur les graphes : le jour porté par le noeud est un jour de fin de semaine ou non. Nous allons dans un premier temps étudier ces deux cas généraux, puis nous verrons ensuite quelques cas particuliers tels que la source, le puits ou les noeuds fictifs (dont l'existence sera justifiée plus loin dans le chapitre).

Cas des noeuds généraux

Deux cas peuvent se présenter lors du parcours du graphe :

- Le jour porté par le noeud considéré est un jour de semaine, auquel cas

on ne sait rien sur la valeur de x_2. Les valeurs acceptables de x_2 sont donc toutes les valeurs comprises entre $-\overline{R2}_{off}$ et $\overline{R2}_{on}$. On aura donc $H = X = [-\overline{R2}_{off}, \overline{R2}_{on}]$.

- Le jour porté par le noeud considéré est un jour de fin de semaine (samedi ou dimanche), auquel cas la valeur minimale de la ressource est 1, puisqu'on est certain que l'infirmière aura au moins travaillé une fin de semaine. On aura donc sur le noeud considéré $H = X = [1, \overline{R2}_{on}]$.

Nous allons maintenant voir les cas particuliers pouvant se présenter dans le cas des noeuds.

Cas de la source

La source peut être fictive ou non. Si elle ne l'est pas, pas de problème puisque le noeud représente la dernière affectation de l'horizon précédent. On se ramène donc au cas général présenté plus haut. Si la source est fictive, c'est-à-dire que l'on ne dispose pas de données concernant l'horizon précédent, on initialise x_2 avec la valeur 0. On aura donc sur une source fictive $H = X = [0, 0]$.

Cas du puits

Le deuxième type de noeud particulier est le puits. Celui-ci étant en fin de graphe et les fenêtres sur les noeuds ne servant qu'à effectuer une mise à jour, ce mêmes fenêtres sont inutiles sur le puits. On aura donc, pour des raisons d'efficacité de l'algorithme, les fenêtres suivantes $H = X = [-\overline{R2}_{off}, \overline{R2}_{on}]$.

Cas des noeuds fictifs

Le dernier type de noeuds particuliers pouvant intervenir dans le graphe est le cas des noeuds fictifs (leurs conditions d'existence seront discutées plus loin dans cette section). Un noeud fictif étant créé (sous certaines conditions) quand un arc modélise une série de fins de semaine prises en congés (on note $\mathcal{N}_{we}$ le nombre de

fins de semaine évitées par l'arc considéré), il faut réinitialiser la valeur de x_2. On aura donc $H = X = [-\mathcal{N}_{we}, -\mathcal{N}_{we}]$ pour forcer la mise à jour de la ressource à $-\mathcal{N}_{we}$.

Récapitulatif

- Noeud général modélisant un jour de semaine : $H = X = [-\overline{R2}_{off}, \overline{R2}_{on}]$.
- Noeud général modélisant un jour de fin de semaine : $H = X = [1, \overline{R2}_{on}]$.
- Cas de la source non virtuelle : en fonction de l'horizon précédent, se rapporter aux cas généraux.
- Cas de la source virtuelle (pas d'horizon précédent) : $H = X = [0, 0]$.
- Cas du puits : $H = X = [-\overline{R2}_{off}, \overline{R2}_{on}]$.
- Cas d'un noeud virtuel modélisant un saut de $\mathcal{N}_{we}$ fins de semaines : $H = X = [-\mathcal{N}_{we}, -\mathcal{N}_{we}]$.

2.4.2.2 Étiquettes portées sur les arcs

Comme on le constatera à la lecture de cette section, de nombreux cas de figures peuvent se présenter pour la gestion de la ressource correspondant aux fins de semaine. Nous allons dans un premier temps étudier les cas correspondant aux cas généraux sans fin de semaine évitée, puis les cas généraux lorsque l'arc indique un ou des fins de semaine prises en congés. Nous terminerons par les arcs particuliers, c'est-à-dire les arcs partant d'une source virtuelle ou bien ceux arrivant au puits.

Lorsqu'un arc évite une ou des fins de semaine, nous noterons $\mathcal{N}_{we}$ ce nombre de fins de semaine prises en congés.

Dans un premier temps, nous allons donc étudier le cas $\mathcal{N}_{we} = 0$, c'est-à-dire que l'arc considéré n'évite aucune fin de semaine. Quatre cas se présentent alors, que nous allons détailler successivement.

Cas $\mathcal{N}_{we} = 0$. Origine : jour de semaine, destination : jour de semaine

Dans ce cas, nous devons autoriser toute valeur de x_2 comprise entre les bornes extrêmes de la ressource, puisqu'aucune modification ne sera faite en empruntant cet arc (les deux jours appartiennent forcément à la même semaine). Nous aurons donc, comme illustré figure 2.3, les données suivantes sur l'arc considéré :

$$\omega = [-\overline{R2}_{off}, \overline{R2}_{on}]; u = 0.$$

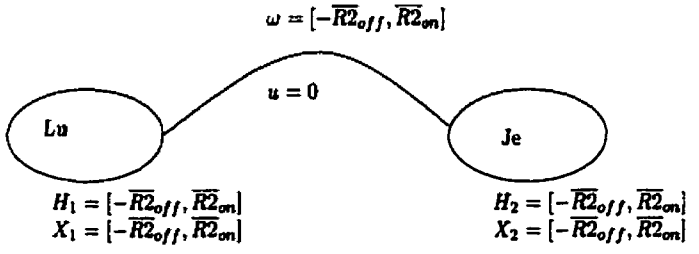


Figure 2.3: Cas semaine à semaine (ressource $R2$)

Cas $\mathcal{N}_{we} = 0$. Origine : jour de fin de semaine, destination : jour de semaine

Dans ce cas, l'arc partant d'un jour de fin de semaine, on se situe dans une série de fins de semaine travaillées. La valeur de la ressource doit donc être comprise entre 1 et le nombre maximum de fins de semaine consécutivement travaillées. Par contre, l'arc aboutissant à un jour de semaine, la consommation de ressource sera là encore égale à zéro. On aura donc (comme illustré figure 2.4), les valeurs suivantes :

$$\omega = [1, \overline{R2}_{on}]; u = 0.$$

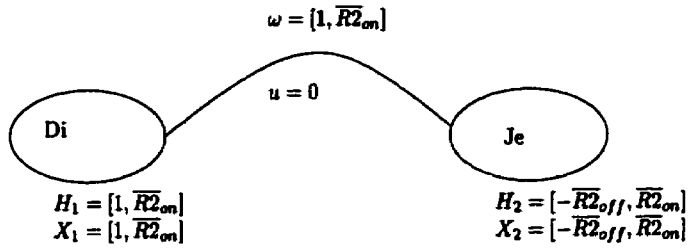


Figure 2.4: Cas fin de semaine à semaine (ressource $R2$)

Cas $\mathcal{N}_{we} = 0$. Origine : jour de semaine, destination : jour de fin de semaine

Ce cas plus complexe est illustré par la figure 2.5. En fait, deux cas de figure peuvent co-exister.

- L'infirmière considérée peut sortir d'une série de fins de semaine prises en congés, auquel cas il faut vérifier que la valeur de la ressource est bien comprise entre les nombres minimum et maximum de fins de semaine successives prises en congés. On aurait donc $\omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]$.
- L'infirmière considérée peut également être dans une série de fins de semaine travaillées, auquel cas on doit vérifier que la ressource est bien comprise entre 0 et le nombre maximum de fin de semaine travaillées à la suite moins une unité. On aurait donc $\omega = [0, \overline{R2}_{on} - 1]$.

Dans tous les cas, l'arc arrivant à un jour de fin de semaine, on aura $u = 1$.

Comme on le voit, deux cas de figure sont possibles dans cette configuration. C'est pourquoi on crée un second arc, "parallèle" au premier, permettant ainsi de représenter les deux cas de figure possibles. On aura donc les données suivantes sur les deux arcs :

$$1^{\text{er}} \text{ arc} : \omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]; u = 1.$$

$$2^{\text{ème}} \text{ arc} : \omega = [0, \overline{R2}_{on} - 1]; u = 1.$$

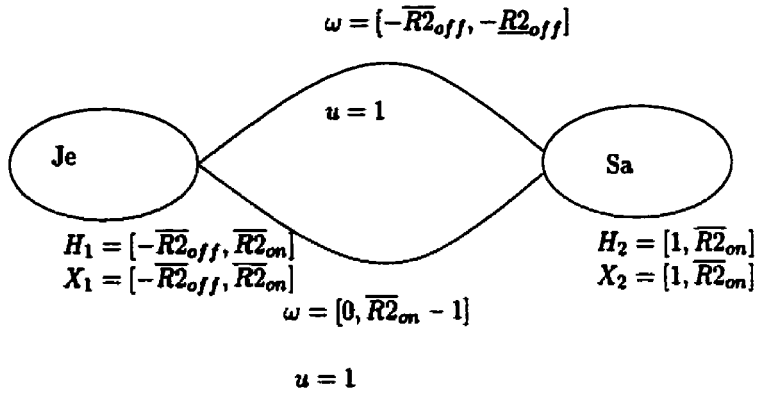


Figure 2.5: Cas semaine à fin de semaine (ressource $R2$)

On verra dans la suite de ce chapitre que ce procédé de dédoublement d'arcs et de création de noeuds virtuels permet de modéliser toutes les configurations possibles quant à la ressource gérant les fins de semaine.

Cas $\mathcal{N}_{we} = 0$. Origine : jour de fin de semaine, destination : jour de fin de semaine

En fait, ce cas logique contient deux configurations possibles.

- Si le jour de départ est un samedi ou un dimanche et que le jour d'arrivée est un samedi ou un dimanche de la semaine suivante (voir figure 2.6), on doit autoriser le passage si la ressource est comprise entre 1 (puisqu'on vient d'effectuer au moins un week-end travaillé) et $\overline{R2}_{on} - 1$ (puisqu'on arrive sur un jour de fin de semaine). La consommation de ressource sera égale à un pour la même raison. On aura donc les données suivantes :

$$\omega = [1, \overline{R2}_{on} - 1]; u = 1.$$

- Si le jour de départ est un samedi et que le jour d'arrivée est le dimanche suivant (donc le lendemain, voir figure 2.7), la consommation de ressource sera nulle (puisqu'on est passé par le samedi, on a déjà compté la consom-

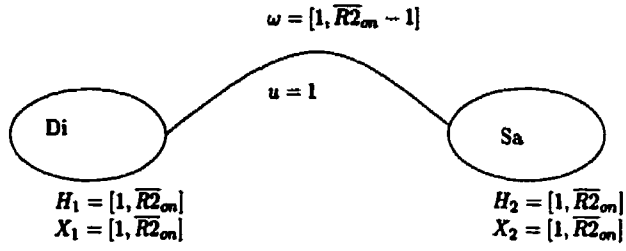


Figure 2.6: Cas fin de semaine à fin de semaine différente (ressource $R2$)

mation de ressource pour cette fin de semaine). Pour la même raison, on peut autoriser le passage sur l'arc si la ressource est comprise entre 1 et le nombre maximum de fin de semaines consécutives travaillées. On aura donc les données suivantes :

$$\omega = [1, \overline{R2}_{on}]; u = 0.$$

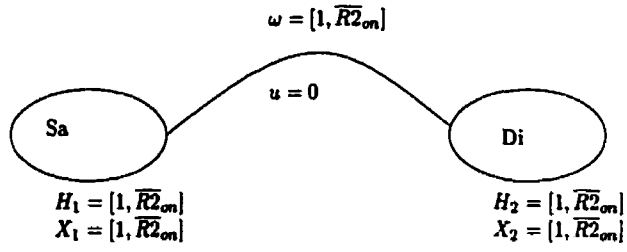


Figure 2.7: Cas fin de semaine à fin de semaine identique (ressource $R2$)

Récapitulatif pour $\mathcal{N}_{we} = 0$

- L'origine représente un jour de semaine, la destination un jour de semaine :

$$\omega = [-\overline{R2}_{off}, \overline{R2}_{on}]; u = 0.$$

- L'origine représente un jour de fin de semaine, la destination un jour de semaine :

$$\omega = [1, \overline{R2}_{on}]; u = 0.$$

- L'origine représente un jour de semaine, la destination un jour de fin de semaine. Il faut réaliser un dédoublement d'arc.

$$1^{\text{er}} \text{ arc} : \omega = [-\overline{R2}_{off}, \underline{R2}_{off}]; u = 1;$$

$$2^{\text{ème}} \text{ arc} : \omega = [0, \overline{R2}_{on} - 1]; u = 1.$$

- L'origine et la destination sont deux jours de fins de semaine situés à une semaine d'intervalle :

$$\omega = [1, \overline{R2}_{on} - 1]; u = 1.$$

- L'origine est un samedi et la destination le dimanche suivant :

$$\omega = [1, \overline{R2}_{on}] \; ; u \; = \; 0.$$

Nous allons maintenant passer en revue les mêmes configurations dans le cas $\mathcal{N}_{we} > 0$. Nous avons donc quatre cas à étudier.

Cas $\mathcal{N}_{we} > 0$. Origine : jour de semaine, destination : jour de semaine

Dans ce cas (figure 2.8), il y a deux configurations possibles :

- On est dans une série de fins de semaine de congés, donc on a en arrivant $x_2 \leq 0$. On doit autoriser le passage si la ressource est comprise entre $-\overline{R2_{off}} + N_{we}$ et 0, puisque l'arc modélise une série de $\mathcal{N}_{we}$ fins de semaine de congés. La consommation de ressource sera de $-\mathcal{N}_{we}$.
- On était dans une série de fins de semaine travaillées, et l'arc modélise le passage à une série de fins de semaine prises en congés. Dans un premier temps, il nous faut vérifier que la série de fins de semaine travaillées avait la bonne longueur. Mais on constate qu'il va falloir également remettre la valeur de la ressource à $-\mathcal{N}_{we}$ pour compter les fins de semaines prises en congés correctement. Pour ce faire, on crée sur ce deuxième arc un noeud

virtuel, qui n'aura d'autre but que la remise à $-\mathcal{N}_{we}$ de la valeur de la ressource. On se retrouve donc avec un arc "coupé" en deux moitiés. Sur la première, on vérifiera la bonne longueur de la série travaillée ; sur la deuxième, on vérifiera la valeur de la ressource avec le nombre de fins de semaine prises en congés.

On aura donc la configuration suivante :

$$1^{er} \text{ arc} : \omega = [-\overline{R2}_{off} + \mathcal{N}_{we}, 0]; u = -\mathcal{N}_{we}.$$

$$2^{\text{ème}} \text{ arc , } 1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ arc , } 2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -1]; u = 0.$$

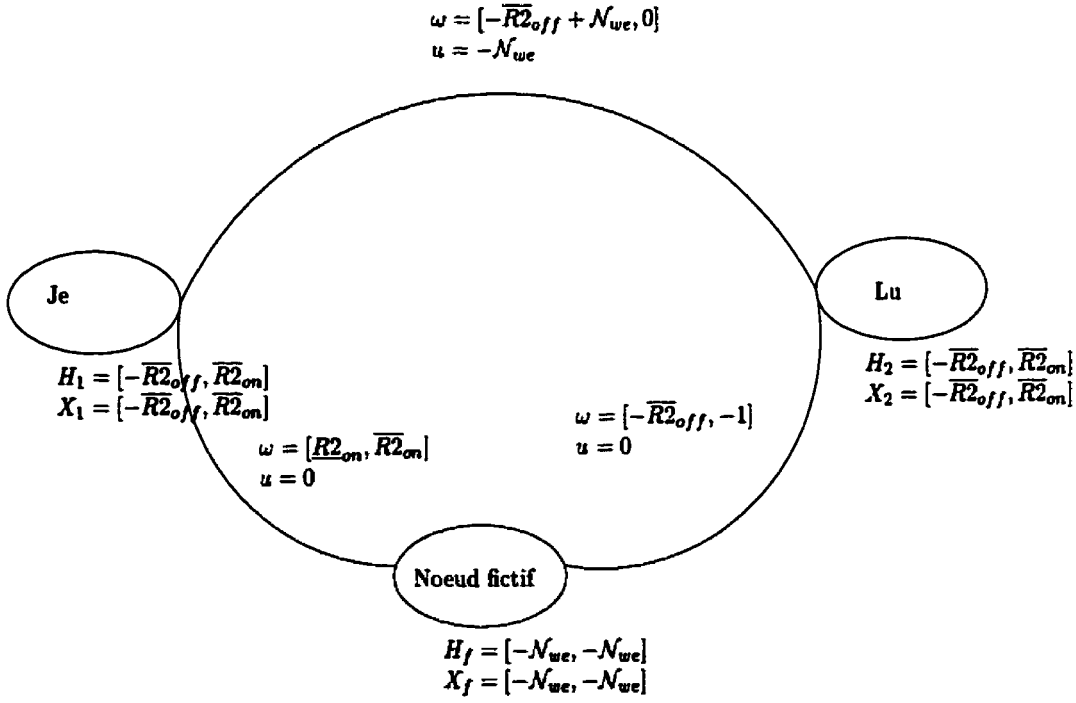


Figure 2.8: Cas semaine à semaine avec $\mathcal{N}_{we} > 0$ (ressource $R2$)

Cas $\mathcal{N}_{we} > 0$. Origine : jour de fin de semaine, destination : jour de semaine

Ce cas est illustré par la figure 2.9. L'arc part d'un jour de fin de semaine, ce qui signifie que l'on est dans une série de fins de semaine travaillées. Par contre, l'arc évite au moins une fin de semaine. On passe donc d'une série travaillée à une série de congés. Il faut donc effectuer plusieurs tâches :

- vérifier que la série travaillée a une longueur acceptable ;
- réinitialiser la valeur de x_2 pour comptabiliser correctement les fins de semaines prises en congés.

La solution sera donc de créer là encore un noeud virtuel sur l'arc considéré. On aura les informations suivantes sur l'arc coupé :

$$1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -1]; u = 0.$$

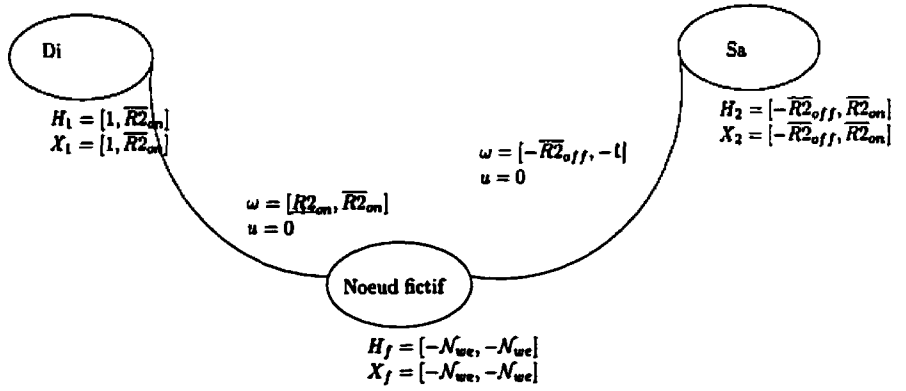


Figure 2.9: Cas fin de semaine à semaine avec $\mathcal{N}_{we} > 0$ (ressource $R2$)

Cas $\mathcal{N}_{we} > 0$. Origine : jour de semaine, destination : jour de fin de semaine

Cette configuration est illustrée par la figure 2.10. Là encore, deux cas sont possibles pour cette configuration, qui vont conduire à un dédoublement d'arc et à la création d'un noeud virtuel sur un des deux arcs. En effet, le noeud de départ représentant un jour de semaine, on n'a aucune information concernant le type de série (travaillée ou non) dans laquelle on se situe pour les fins de semaine. Les deux cas sont donc comme précédemment les suivants :

- On arrive au noeud de départ avec $x_2 \leq 0$. On est donc dans une série de congés. L'arc considéré évitant plusieurs fins de semaine, on reste dans cette série de congés. Par contre, l'arc arrivant sur un jour de fin de semaine, la série de fins de semaine prises en congés se termine. Il faut donc vérifier que le nombre de congés est correct. On aura donc, pour cet arc, $u = -\mathcal{N}_{we}$ et $\omega = [-\overline{R2}_{off} + \mathcal{N}_{we}, -\underline{R2}_{off} + \mathcal{N}_{we}]$.

  Note : Dans ce cas précis, la valeur de la consommation u est inutile car la ressource sera remise à jour à l'arrivée sur le noeud cible qui représente un jour de fin de semaine (voir la section précédente).

- On arrive au noeud de départ avec $x_2 > 0$. On est donc dans une série de fins de semaine travaillées. L'arc considéré évitant certaines fins de semaine, on passe à une série de fins de semaine non travaillées. Puis, le noeud d'arrivée étant un jour de fin de semaine, on repasse à une série de fins de semaine travaillées. Il faut donc vérifier que le nombre de fins de semaine travaillées précédemment était correct, puis remettre la valeur de la ressource à zéro, et enfin compter le nombre de fins de semaine prises en congés modélisées par l'arc considéré. On va donc créer un noeud fictif. Sur la première moitié de l'arc, on aura $u = 0$ et $\omega = [\underline{R2}_{on}, \overline{R2}_{on}]$. Sur la seconde moitié, on aura $u = 0$ et $\omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]$. La valeur de la ressource étant négative,

elle sera automatiquement remise à 1 lors de l'arrivée sur le noeud cible.

Pour résumer, les données sont donc les suivantes dans ce cas de figure :

$$1^{\text{er}} \text{ arc} : \omega = [-\overline{R2}_{off} + \mathcal{N}_{we}, -\underline{R2}_{off} + \mathcal{N}_{we}]; u = -\mathcal{N}_{we}.$$

$$2^{\text{ème}} \text{ arc}, 1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ arc}, 2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]; u = 0.$$

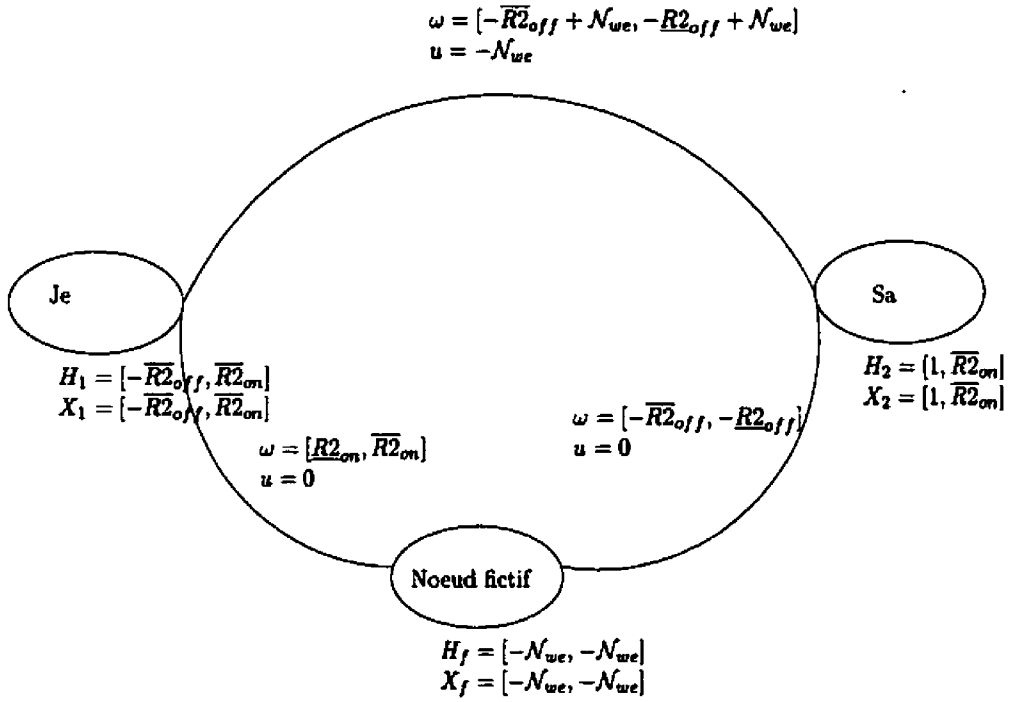


Figure 2.10: Cas semaine à fin de semaine avec $\mathcal{N}_{we} > 0$ (ressource $R2$)

Cas $\mathcal{N}_{we} > 0$. Origine : jour de fin de semaine, destination : jour de fin de semaine

Ce cas est représenté figure 2.11. Le noeud source représentant un jour de fin de semaine, on sort d'une série de fins de semaine travaillées. Il faut donc dans un premier temps vérifier la validité de la longueur de cette série, puis remettre la

ressource au nombre de fins de semaine non travaillées (rôle du noeud fictif), et enfin vérifer la validité du nombre de fins de semaine non travaillées. On aura donc les informations suivantes :

$$1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]; u = 0.$$

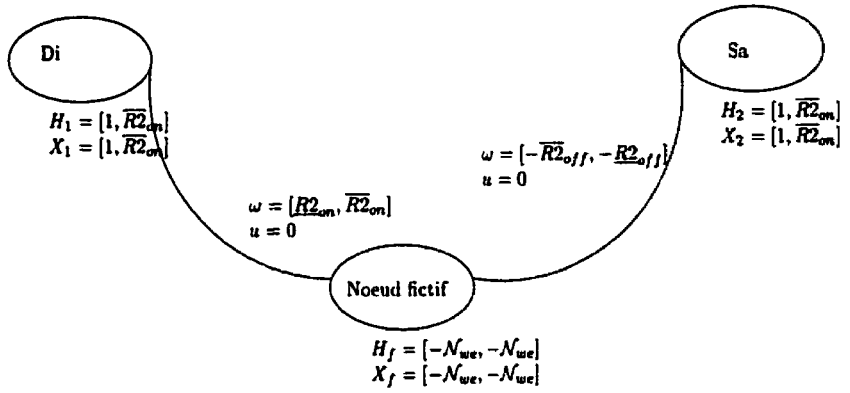


Figure 2.11: Cas fin de semaine à fin de semaine avec $\mathcal{N}_{we} > 0$ (ressource $R2$)

Récapitulatif pour $\mathcal{N}_{we} > 0$

- L'origine représente un jour de semaine, la destination un jour de semaine. Il faut dédoubler l'arc et créer un noeud virtuel sur une des copies.

$$1^{\text{er}} \text{ arc} : \omega = [-\overline{R2}_{off} + \mathcal{N}_{we}, 0]; u = -\mathcal{N}_{we}.$$

$$2^{\text{ème}} \text{ arc}, 1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ arc}, 2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -1]; u = 0.$$

- L'origine représente un jour de fin de semaine, la destination un jour de semaine. Il faut créer un noeud virtuel sur l'arc considéré.

$$1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -1]; u = 0.$$

- L'origine représente un jour de semaine, la destination un jour de fin de semaine. Il faut dédoubler l'arc et créer un noeud virtuel sur une des copies.

$$1^{\text{er}} \text{ arc} : \omega = [-\overline{R2}_{off} + \mathcal{N}_{we}, -\underline{R2}_{off} + \mathcal{N}_{we}]; u = -\mathcal{N}_{we}.$$

$$2^{\text{ème}} \text{ arc}, 1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ arc}, 2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]; u = 0.$$

- L'origine et la destination sont deux jours de fins de semaine. Il faut créer un noeud virtuel.

$$1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]; u = 0.$$

Nous avons donc terminé l'étude des différents cas qui peuvent se présenter pour les arcs dits généraux. Nous allons maintenant nous pencher sur le problème des arcs particuliers, c'est-à-dire les arcs partant de la source ou arrivant au noeud puits.

Cas des arcs partant de la source

Si la source est un noeud explicite, c'est-à-dire si l'on dispose des informations concernant l'horizon précédent, pas de problème particulier puisque la source représente dans ce cas la dernière affectation de l'horizon précédent. On se ramène alors au cas général. Les problèmes surviennent lorsque la source est un noeud fictif. Dans ce cas, on démarre le parcours du graphe avec $x_2 = 0$. Deux cas peuvent alors se présenter à nous :

- L'arc considéré arrive sur un jour de fin de semaine. Qu'il y ait ou non des fins de semaine non travaillées entre le premier jour de l'horizon et la cible de l'arc, les données seront les mêmes. En effet, même s'il y a des fins de

semaine non travaillées, l'existence même de l'arc prouve que leur nombre est compatible avec les contraintes dures. On aura donc une consommation de ressources de 1, puisque le noeud cible est un noeud de fin de semaine. On aura donc les données suivantes sur l'arc :

$$\omega = [0, 0]; u = 1.$$

- L'arc considéré arrive sur un jour de semaine. Là encore, si on note $\mathcal{N}_{we}$ le nombre de fins de semaine entiers entre le premier jour de l'horizon et le jour représenté par le noeud cible, il suffit de mettre à jour la valeur de la ressource avec ce nombre de fins de semaine prises en congés. On aura donc les données suivantes sur l'arc :

$$\omega = [0, 0]; u = -\mathcal{N}_{we}.$$

Cas des arcs arrivant au puits

Ce dernier noeud est toujours virtuel. On note alors $\mathcal{N}_{we}$ le nombre de fins de semaine comprises entre le jour représenté par le noeud origine et le lendemain du dernier jour de l'horizon courant. Là encore, plusieurs configurations peuvent se présenter :

- Si le noeud origine est un jour de fin de semaine et que $\mathcal{N}_{we} = 0$: on sort d'une série de fins de semaine travaillées, il faut donc vérifier que sa longueur respecte les contraintes dures. On aura donc les données suivantes :

$$\omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

- Si le noeud origine est un jour de semaine et que $\mathcal{N}_{we} = 0$, deux cas peuvent se présenter, il va donc falloir dédoubler l'arc. Le premier servira à autoriser le passage si l'infirmière sort d'une série de fins de semaine travaillées de longueur correcte. Le deuxième arc servira à vérifier la validité si l'infirmière

sort d'une série de fins de semaine prises en congés. On aura donc les données suivantes sur les arcs :

$$1^{\text{er}} \text{ arc} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ arc} : \omega = [-\overline{R2}_{off}, \underline{R2}_{off}]; u = 0.$$

- Si le noeud origine est un jour de fin de semaine et que $\mathcal{N}_{we} > 0$, on sort d'une série de fins de semaine travaillées, puis on réalise une série de fins de semaine non travaillées. Il faut donc créer un noeud virtuel sur cet arc. Ce noeud virtuel servira à réinitialiser la valeur de x_2 avec le nombre de fins de semaine prises en congés. On aura donc les données suivantes :

$$1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]; u = 0.$$

- Si le noeud origine représente un jour de semaine et que $\mathcal{N}_{we} > 0$, on retrouve la discussion classique car deux configurations peuvent se présenter : soit on est dans une série de fins de semaines non travaillées et on ne fait qu'en rajouter, soit on sort d'une série de fins de semaines travaillées et on termine par une série de fins de semaine prises en congés. On retrouve donc un dédoublement d'arc, avec présence d'un noeud virtuel sur l'arc correspondant au changement de situation. On a donc les données suivantes :

$$1^{\text{er}} \text{ arc} : \omega = [-\overline{R2}_{off} + \mathcal{N}_{we}, -\underline{R2}_{off} + \mathcal{N}_{we}]; u = 0.$$

$$2^{\text{ème}} \text{ arc}, 1^{\text{ere}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ arc}, 2^{\text{eme}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]; u = 0.$$

Ceci conclut donc l'étude des différentes configurations possibles pour les arcs en ce qui concerne la deuxième ressource.

Récapitulatif pour les arcs particuliers

- Cas où l'origine est la source (virtuelle), la destination un jour de fin de semaine :

$$\omega = [0, 0]; u = 1.$$

- Cas où l'origine est la source (virtuelle), la destination un jour de semaine :

$$\omega = [0, 0]; u = 1.$$

- Cas où l'origine est un jour de fin de semaine, la destination le puits et $\mathcal{N}_{we} = 0$.

$$\omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

- Cas où l'origine est un jour de semaine, la destination le puits et $\mathcal{N}_{we} = 0$. On a un dédoublement d'arc.

$$1^{\text{er}} \text{ arc} : \omega = [-\overline{R2}_{off}, \underline{R2}_{off}]; u = 0.$$

$$2^{\text{ème}} \text{ arc} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

- Cas où l'origine est un jour de fin de semaine, la destination le puits et $\mathcal{N}_{we} > 0$. On a création d'un noeud virtuel.

$$1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -\underline{R2}_{off}, 0]; u = 0.$$

- Cas où l'origine est un jour de semaine, la destination le puits et $\mathcal{N}_{we} > 0$. On a dédoublement de l'arc et création d'un noeud virtuel.

$$1^{\text{er}} \text{ arc} : \omega = [-\overline{R2}_{off} + \mathcal{N}_{we}, -\underline{R2}_{off} + \mathcal{N}_{we}]; u = 0.$$

$$2^{\text{ème}} \text{ arc}, 1^{\text{ère}} \text{ moitié} : \omega = [\underline{R2}_{on}, \overline{R2}_{on}]; u = 0.$$

$$2^{\text{ème}} \text{ arc}, 2^{\text{ème}} \text{ moitié} : \omega = [-\overline{R2}_{off}, -\underline{R2}_{off}]; u = 0.$$

2.5 Ressource associée à la rotation des types de quarts

2.5.1 Définitions et hypothèses préalables

Quelque soit le nombre de quarts de travail retenus par les hôpitaux, on peut toujours regrouper ces quarts de travail en trois types : quarts de jour, de soir ou de nuit. La troisième ressource utilisée par le problème, notée $R3$ dans la suite du document, va nous permettre de contrôler certaines règles associées à ces types de quart. En fait, on veut contrôler le nombre minimal et maximal de quarts de chaque type effectués consécutivement.

On va donc disposer des données suivantes :

- $\underline{\mathcal{D}}$ et $\overline{\mathcal{D}}$, les nombres minimum et maximum de quarts de type "jour" consécutifs;
- $\underline{\mathcal{E}}$ et $\overline{\mathcal{E}}$, les nombres minimum et maximum de quarts de type "soir" consécutifs;
- $\underline{\mathcal{N}}$ et $\overline{\mathcal{N}}$, les nombres minimum et maximum de quarts de type "nuit" consécutifs.

On notera dans la suite de cette section x_3 la valeur courante du compteur associé à cette troisième ressource. On adopte alors la modélisation suivante :

- Chaque quart traversé incrémente la valeur de la ressource de 1 unité.
- Si $1 \leq x_3 \leq \overline{\mathcal{D}}$, on est dans une série de quarts de jour.
- Si $\overline{\mathcal{D}}+1 \leq x_3 \leq \overline{\mathcal{D}}+\overline{\mathcal{E}}$, on est dans une série de quarts de soir.
- Si $\overline{\mathcal{D}}+\overline{\mathcal{E}}+1 \leq x_3 \leq \overline{\mathcal{D}}+\overline{\mathcal{E}}+\overline{\mathcal{N}}$, on est dans une série de quarts de nuit.
- A chaque changement de série, on vérifie la validité en terme de longueur de la série courante et on réinitialise la valeur de la ressource au premier élément de la fenêtre correspondant au nouveau type de quart.

2.5.2 Étude des différents cas

Nous allons commencer par étudier les différentes fenêtres de mise à jour dont nous avons besoin pour les noeuds. Comme pour les autres ressources, nous commencerons par les noeuds généraux du graphe pour finir par les noeuds particuliers.

2.5.2.1 Étiquettes sur les noeuds

Cas des noeuds généraux

Pour les noeuds généraux, le critère discriminant pour choisir la fenêtre de mise à jour portée par le noeud est le type de quart de travail que ce noeud représente. Trois cas sont alors possibles.

- Si le noeud représente un quart de travail de type "jour", alors la fenêtre de mise à jour sera $H = X = [1, \overline{\mathcal{D}}]$. En effet, en passant par ce noeud, on s'assure avoir réalisé au moins un quart de type "jour". Les contraintes sur les arcs menant à ce noeud permettent de s'assurer quant à elles que le nombre de quarts de types jour consécutifs reste bien inférieur à $\overline{\mathcal{D}}$.
- Si le noeud représente un quart de type "soir", on aura les fenêtres suivantes : $H = X = [\overline{\mathcal{D}} + 1, \overline{\mathcal{D}} + \overline{\mathcal{E}}]$.
- Si le noeud représente un quart de type "nuit", on aura les fenêtres de mise à jour : $H = X = [\overline{\mathcal{D}} + \overline{\mathcal{E}} + 1, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{N}}]$.

Cas de la source

Si on dispose de l'horizon précédent, donc si la source n'est pas virtuelle, on applique les règles précédentes. Si au contraire la source est un noeud virtuel, on aura : $H = X = [0, 0]$.

Cas du puits

Pour le puits, les fenêtres de mise à jour n'ont que peu d'importance. On peut donc poser $H = X = [0, 0]$.

Cas des noeuds fictifs

Pour les noeuds fictifs résultant de cas particuliers dans la gestion de la ressource liée aux fins de semaine, la fenêtre de mise à jour pour la troisième ressource n'a là encore que peu d'importance. On pose alors : $H = X = [-\infty, +\infty]$.

Récapitulatif

- Noeud représentant un quart de travail de type jour : $H = X = [1, \overline{\mathcal{D}}]$.
- Noeud représentant un quart de travail de type soir : $H = X = [\overline{\mathcal{D}}+1, \overline{\mathcal{D}}+\overline{\mathcal{E}}]$.
- Noeud représentant un quart de travail de type nuit : $H = X = [\overline{\mathcal{D}} + \overline{\mathcal{E}} + 1, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{N}}]$.
- Source non virtuelle (historique présent) : voir cas précédents.
- Source virtuelle (historique absent) : $H = X = [0, 0]$.
- Puits : $H = X = [0, 0]$.
- Noeud virtuel : $H = X = [-\infty, +\infty]$.

2.5.2.2 Étiquettes sur les arcs

Comme pour les autres ressources, nous allons diviser les arcs en deux catégories : nous commencerons par étudier les arcs dits généraux, c'est-à-dire les arcs joignant deux noeuds non particuliers, puis nous terminerons par les arcs partant de la source, arrivant au puits, ou passant par un noeud fictif.

Dans le cas des arcs généraux, trois cas peuvent se présenter qui vont structurer notre étude.

Cas où le noeud origine représente un quart de travail de type jour

Ce cas est illustré par la figure 2.12. Dans cette configuration, le noeud origine représentant un quart de type "jour", on est dans une série de quarts de jour. Si la cible de l'arc est un quart de type jour, il faut vérifier que l'on peut ajouter un quart de type jour à la série courante. Si la cible est un quart de type "soir" ou "nuit", il faut vérifier le validité de la longueur de la série que l'on vient de réaliser. Pour les consommations, la valeur est d'une unité, si on continue dans la série de quarts de type jour, et d'une valeur forçant la remise à zéro sinon : en effet, la remise à niveau de la valeur de la ressource x_3 se fera lors de l'arrivée sur le noeud destination avec les fenêtres de mise à jour. Pour résumer, les informations portées par les arcs seront donc les suivantes :

$$\text{Origine : jour - Destination : jour} \rightarrow \omega = [1, \overline{\mathcal{D}} - 1]; u = 1.$$

$$\text{Origine : jour - Destination : soir} \rightarrow \omega = [\underline{\mathcal{D}}, \overline{\mathcal{D}}]; u = -\overline{\mathcal{D}}.$$

$$\text{Origine : jour - Destination : nuit} \rightarrow \omega = [\underline{\mathcal{D}}, \overline{\mathcal{D}}]; u = -\overline{\mathcal{D}}.$$

Cas où le noeud origine représente un quart de travail de type soir

Ce cas est illustré par la figure 2.13. Dans cette configuration, le noeud origine représentant un quart de type "soir", on est dans une série de quarts de soir. Si la cible de l'arc est un quart de type soir, il faut vérifier que l'on peut ajouter un quart de type soir à la série courante. Si la cible est un quart de type "jour" ou "nuit", il faut vérifier le validité de la longueur de la série que l'on vient de réaliser. Pour résumer, les informations portées par les arcs seront donc les suivantes :

$$\text{Origine : soir - Destination : jour} \rightarrow \omega = [\overline{\mathcal{D}} + \underline{\mathcal{E}}, \overline{\mathcal{D}} + \overline{\mathcal{E}}]; u = -\overline{\mathcal{D}} - \overline{\mathcal{E}}.$$

$$\text{Origine : soir - Destination : soir} \rightarrow \omega = [\overline{\mathcal{D}} + 1, \overline{\mathcal{D}} + \overline{\mathcal{E}} - 1]; u = 1.$$

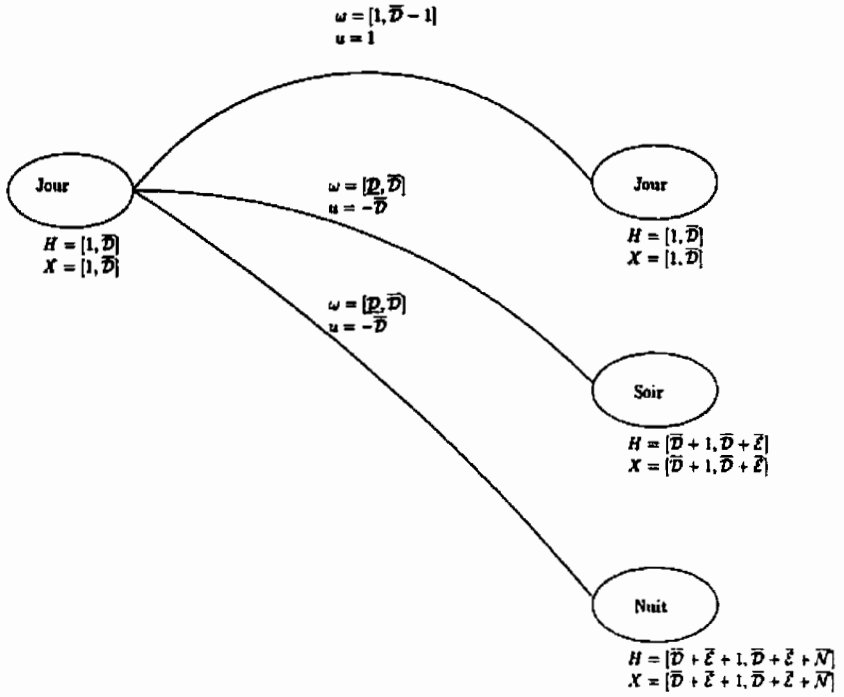


Figure 2.12: Cas noeud de départ de type "jour" (ressource $R3$)

Origine : soir - Destination : nuit $\rightarrow \omega = [\overline{\mathcal{D}} + \underline{\mathcal{E}}, \overline{\mathcal{D}} + \overline{\mathcal{E}}]; u = -\overline{\mathcal{D}} - \overline{\mathcal{E}}.$

Cas où le noeud origine représente un quart de travail de type nuit

Ce cas est illustré par la figure 2.14. Dans cette configuration, le noeud origine représentant un quart de type "nuit", on est dans une série de quarts de nuit. Si la cible de l'arc est un quart de type nuit, il faut vérifier que l'on peut ajouter un quart de type nuit à la série courante. Si la cible est un quart de type "jour" ou "soir", il faut vérifier la validité de la longueur de la série que l'on vient de réaliser. Pour résumer, les informations portées par les arcs seront donc les suivantes :

Origine : nuit - Destination : jour $\rightarrow \omega = [\overline{\mathcal{D}}+\overline{\mathcal{E}}+\underline{\mathcal{N}}, \overline{\mathcal{D}}+\overline{\mathcal{E}}+\overline{\mathcal{N}}]; u = -\overline{\mathcal{D}}-\overline{\mathcal{E}}-\overline{\mathcal{N}}.$

Origine : nuit - Destination : soir $\rightarrow \omega = [\overline{\mathcal{D}}+\overline{\mathcal{E}}+\underline{\mathcal{N}}, \overline{\mathcal{D}}+\overline{\mathcal{E}}+\overline{\mathcal{N}}]; u = -\overline{\mathcal{D}}-\overline{\mathcal{E}}-\overline{\mathcal{N}}.$

Origine : nuit - Destination : nuit $\rightarrow \omega = [\overline{\mathcal{D}} + \overline{\mathcal{E}} + 1, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{N}} - 1]; u = 1.$

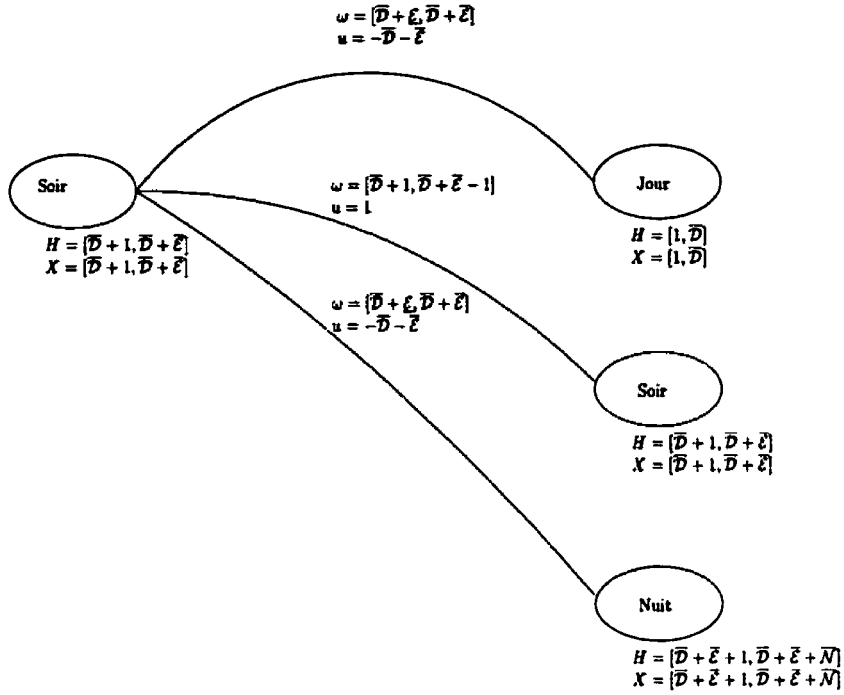


Figure 2.13: Cas noeud de départ de type "soir" (ressource $R3$)

Nous allons maintenant voir les cas particuliers pouvant intervenir dans le cadre de la gestion de la ressource $R3$ sur les arcs.

Cas d'un arc dont le noeud origine est la source

Si le noeud source est explicite, c'est-à-dire si on dispose des informations concernant l'horizon précédent, alors on se rapporte aux cas généraux vus précédemment.

Si la source est virtuelle, trois cas se présentent.

- Le noeud destination représente un quart de travail de type "jour". On aura les données suivantes :

$$\omega = [0, 0]; u = 1.$$

- Le noeud destination représente un quart de travail de type "soir". On aura

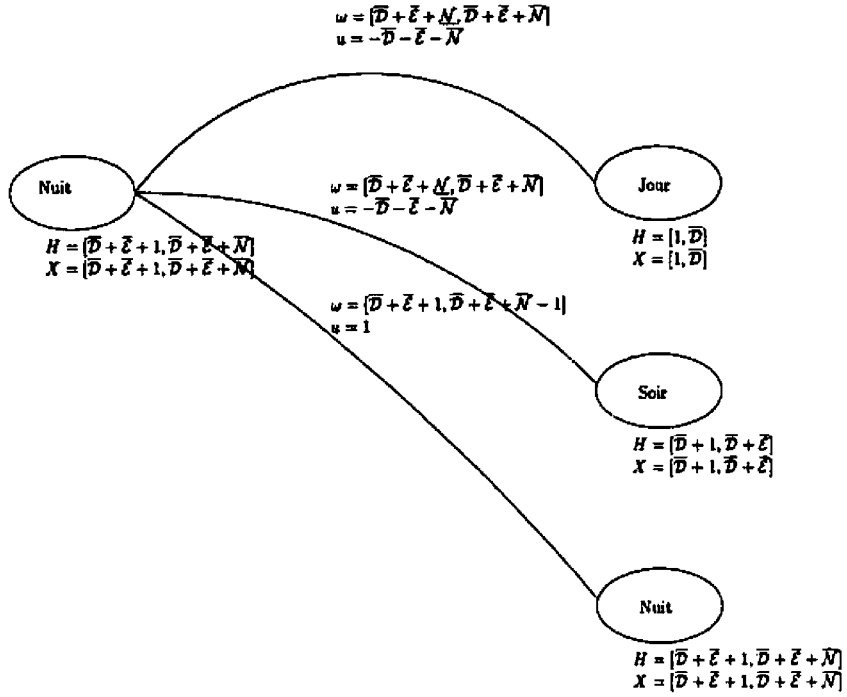


Figure 2.14: Cas noeud de départ de type "nuit" (ressource $R3$)

les données suivantes :

$$\omega = [0, 0]; u = \overline{\mathcal{D}} + 1.$$

- Le noeud destination représente un quart de travail de type "nuit". On aura les données suivantes :

$$\omega = [0, 0]; u = \overline{\mathcal{D}} + \overline{\mathcal{E}} + 1.$$

Cas d'un arc dont le noeud destination est le puits

Pour le puits, celui-ci étant toujours virtuel, il nous faut vérifier la validité de la série terminale. Là encore, trois cas se présentent.

- Le noeud origine de l'arc représente un quart de travail de type "jour". On aura les données suivantes sur l'arc :

$$\omega = [\underline{\mathcal{D}}, \overline{\mathcal{D}}]; u = 0.$$

- Le noeud origine de l'arc représente un quart de travail de type "soir". On aura les données suivantes sur l'arc :

$$\omega = [\overline{\mathcal{D}} + \underline{\mathcal{E}}, \overline{\mathcal{D}} + \overline{\mathcal{E}}]; u = 0.$$

- Le noeud origine de l'arc représente un quart de travail de type "nuit". On aura les données suivantes sur l'arc :

$$\omega = [\overline{\mathcal{D}} + \overline{\mathcal{E}} + \underline{\mathcal{N}}, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{N}}]; u = 0.$$

Cas d'un arc passant par un noeud virtuel

Pour le cas d'un noeud fictif présent sur un arc (v_{ℓ_1}, v_{ℓ_2}), la situation est simple : le noeud fictif ne servant que pour la ressource $R2$, un formalisme est adopté. Sur la première moitié de l'arc, on fait figurer un test non contraignant. Sur la deuxième moitié, on fait figurer les informations qui auraient normalement été présentes sur l'arc (v_{ℓ_1}, v_{ℓ_2}). On retrouve donc les étiquettes suivantes :

$$1^{\text{ère}} \text{ moitié} : \omega = [0, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{D}}]; u = 0.$$

$2^{\text{ème}}$ moitié : données présentes s'il n'y avait pas de noeud fictif sur (v_{ℓ_1}, v_{ℓ_2}).

2.5.2.3 Récapitulatif

- Noeud origine : jour ; noeud destination : jour.

$$\omega = [1, \overline{\mathcal{D}} - 1]; u = 1.$$

- Noeud origine : jour ; noeud destination : soir.

$$\omega = [\underline{\mathcal{D}}, \overline{\mathcal{D}}]; u = -\overline{\mathcal{D}}.$$

- Noeud origine : jour ; noeud destination : nuit.

$$\omega = [\underline{\mathcal{D}}, \overline{\mathcal{D}}]; u = -\overline{\mathcal{D}}.$$

- Noeud origine : soir ; noeud destination : jour.

$$\omega = [\overline{\mathcal{D}} + \underline{\mathcal{E}}, \overline{\mathcal{D}} + \overline{\mathcal{E}}]; u = -\overline{\mathcal{D}} - \overline{\mathcal{E}}.$$

- Noeud origine : soir ; noeud destination : soir.

$$\omega = [\overline{\mathcal{D}} + 1, \overline{\mathcal{D}} + \overline{\mathcal{E}} - 1]; u = 1.$$

- Noeud origine : soir ; noeud destination : nuit.

$$\omega = [\overline{\mathcal{D}} + \underline{\mathcal{E}}, \overline{\mathcal{D}} + \overline{\mathcal{E}}]; u = -\overline{\mathcal{D}} - \overline{\mathcal{E}}.$$

- Noeud origine : nuit ; noeud destination : jour.

$$\omega = [\overline{\mathcal{D}} + \overline{\mathcal{E}} + \underline{\mathcal{N}}, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{N}}]; u = -\overline{\mathcal{D}} - \overline{\mathcal{E}} - \overline{\mathcal{N}}.$$

- Noeud origine : nuit ; noeud destination : soir.

$$\omega = [\overline{\mathcal{D}} + \overline{\mathcal{E}} + \underline{\mathcal{N}}, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{N}}]; u = -\overline{\mathcal{D}} - \overline{\mathcal{E}} - \overline{\mathcal{N}}.$$

- Noeud origine : nuit ; noeud destination : nuit.

$$\omega = [\overline{\mathcal{D}} + \overline{\mathcal{E}} + 1, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{N}} - 1]; u = 1.$$

- Noeud origine : source non virtuelle : se reporter aux cas précédents.
- Noeud origine : source virtuelle ; noeud destination : jour.

$$\omega = [0, 0]; u = 1.$$

- Noeud origine : source virtuelle ; noeud destination : soir.

$$\omega = [0, 0]; u = \overline{\mathcal{D}} + 1.$$

- Noeud origine : source virtuelle ; noeud destination : nuit.

$$\omega = [0, 0]; u = \overline{\mathcal{D}} + \overline{\mathcal{E}} + 1.$$

- Noeud origine : jour ; noeud destination : puits.

$$\omega = [\underline{\mathcal{D}}, \overline{\mathcal{D}}]; u = 0.$$

- Noeud origine : soir ; noeud destination : puits.

$$\omega = [\overline{\mathcal{D}} + \underline{\mathcal{E}}, \overline{\mathcal{D}} + \overline{\mathcal{E}}]; u = 0.$$

- Noeud origine : nuit ; noeud destination : puits.

$$\omega = [\overline{\mathcal{D}} + \overline{\mathcal{E}} + \underline{\mathcal{N}}, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{N}}]; u = 0.$$

- Noeud virtuel sur l'arc.

$$1^{\text{ère}} \text{ moitié} : \omega = [0, \overline{\mathcal{D}} + \overline{\mathcal{E}} + \overline{\mathcal{D}}]; u = 0.$$

$2^{\text{ème}}$ moitié : données présentes sur l'arc(v_{ℓ_1}, v_{ℓ_2}) sans noeud fictif.

2.6 Ressource associée aux jours de congés

2.6.1 Définitions et hypothèses préalables

Il existe dans notre problème deux types de journées non travaillées pour une infirmière :

- les journées dites de repos, qui sont non payées pour l'infirmière;
- les jours de congés, qui sont payés sur un nombre d'heures défini par chaque hôpital.

Comme dans toute profession, le nombre de jours de congés pour une infirmière est limité et doit être contrôlé. C'est le rôle de la ressource $R4$, qui va comptabiliser le nombre de jours de congés accordés à une infirmière sur l'horizon courant. Nous disposons pour cela de données qui sont les suivantes :

- les jours extrèmes entre lesquels l'infirmière est autorisée à prendre ses congés ;
- les nombres minimaux et maximaux de jours de congés que l'infirmière est autorisée à prendre pendant l'horizon courant. On notera $\underline{R4}$ et $\overline{R4}$ ces bornes ;
- une liste de jours candidats que l'infirmière peut prendre en congés. Si cette liste est vide, on considère que tous les jours situés entre le jour au plus tôt et le jour au plus tard sont candidats pour être accordés congés.

De plus, on notera $R4(D_{i_1}, D_{i_2})$ le nombre de jours candidats situés entre le lendemain de D_{i_1} et le jour D_{i_2}.

2.6.2 Étude des différents cas

Dans cette section, nous verrons tout d'abord les étiquettes portées par les noeuds. Comme pour les autres ressources, nous commencerons par les noeuds généraux du graphe, puis nous enchaînerons sur les noeuds particuliers (noeuds fictifs, source et puits).

2.6.2.1 Étude des étiquettes sur les noeuds

Pour cette ressource, on peut diviser les noeuds généraux du graphe en trois catégories : les noeuds situés avant le jour au plus tôt, les noeuds situés entre les jours extrêmes et les jours situés après le jour au plus tard. Nous allons donc étudier successivement ces trois cas de figure.

Cas des noeuds représentant des jours situés avant le premier jour où on peut accorder un congé

La ressource peut être comprise entre 0 (on n'a pas encore assigné de jours de congés) et $\overline{R4}$, puisqu'on peut hériter des jours de congés de l'horizon précédent.

On aura donc $H = X = [0, \overline{R4}]$.

Cas des noeuds représentant des jours situés entre les deux dates extrêmes

La ressource peut là encore être comprise entre 0 et le nombre maximum de jours que l'on peut accorder en congés. On aura donc là encore : $H = X = [0, \overline{R4}]$.

Cas des noeuds représentant des jours situés après le dernier jour où l'on peut octroyer un congé

La ressource sera forcément comprise entre les nombres minimaux et maximaux de jours de congés à accorder. On aura donc sur ce type de noeuds : $H = X = [\underline{R4}, \overline{R4}]$.

Dans les cas des noeuds spéciaux, trois cas se présentent là encore.

Cas des noeuds virtuels

Ceux-ci ne servent que pour la ressource contrôlant les fins de semaine. Ils n'ont aucun rôle à jouer dans le cas de la contrainte de congés. On aura donc sur ce type de noeuds : $H = X = [-\infty, +\infty]$.

Cas de la source

La ressource vaut normalement 0, sauf si on décide de tenir compte des congés accordés au cours de l'horizon précédent. On aura donc sur le noeud source les données suivantes : $H = X = [0, \overline{R4}]$.

Cas du puits

La ressource doit être comprise entre les deux bornes affectées pour le contrôle du nombre de jours de congés accordés. On aura donc sur le puits : $H = X = [\underline{R4}, \overline{R4}]$.

Récapitulatif

- Cas des noeuds représentant des jours situés avant le premier jour que l'on peut accorder congés : $H = X = [0, \overline{R4}]$.
- Cas des noeuds représentant des jours situés entre les deux dates extrêmes : $H = X = [0, \overline{R4}]$.
- Cas des noeuds représentant des jours situés après le dernier jour que l'on peut assigner congés : $H = X = [\underline{R4}, \overline{R4}]$.
- Cas des noeuds virtuels : $H = X = [-\infty, +\infty]$.
- Cas de la source : $H = X = [0, \overline{R4}]$.
- Cas du puits : $H = X = [\underline{R4}, \overline{R4}]$.

2.6.2.2 Étiquettes sur les arcs

La gestion de cette ressource est particulièrement facile sur les arcs "généraux" du graphe. En effet, celle-ci se résume à l'étude de deux cas de figure possibles.

Cas où le noeud cible porte un jour qui se situe avant (ou qui est) le dernier jour où l'on peut accorder un congé

Ce cas est illustré par la figure 2.15. Si D_{i_2} (le jour porté par le sommet cible de l'arc considéré) se situe avant le (ou est égal au) dernier jour pour lequel on peut affecter un quart de congés, on doit autoriser le passage tant que la valeur de la ressource est inférieure au nombre maximum de jours de congés que l'on peut affecter. De plus, la consommation de ressource sera égale à $R4(D_{i_1}, D_{i_2})$. On aura donc, sur l'arc considéré :

$$\omega = [0, \overline{R4} - R4(D_{i_1}, D_{i_2})]; u = R4(D_{i_1}, D_{i_2}).$$

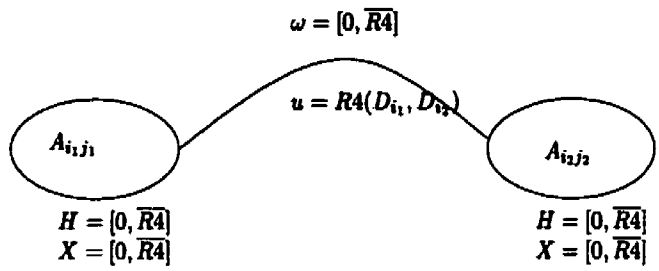


Figure 2.15: Cas noeud cible avant dernier jour candidat pour congés (ressource $R4$)

Cas où le noeud cible porte un jour qui se situe après le dernier jour où l'on peut accorder un congé

Ce cas est représenté par la figure 2.16. Si D_{i_2} se situe après le dernier jour que l'on peut choisir pour affecter une journée de congé, il faut contrôler que l'on a affecté un nombre correct de jours de congés. Pour ce faire, il faut que la valeur courante de la ressource au franchissement de cet arc soit comprise entre $\underline{R4}$ et $\overline{R4}$. Par contre, on peut laisser la même formule de mise à jour de la valeur de la ressource. On aura donc les données suivantes sur l'arc considéré :

$$\omega = [\underline{R4} - R4(D_{i_1}, D_{i_2}), \overline{R4} - R4(D_{i_1}, D_{i_2})]; u = R4(D_{i_1}, D_{i_2}).$$

Cas d'un arc passant par un noeud virtuel

On aura les données suivantes :

$$1^{\text{ère}} \text{ moitié} : \omega = [0, \overline{R4}]; u = 0.$$

$2^{\text{ème}}$ moitié : données présentes s'il n'y avait pas de noeud fictif sur (v_{ℓ_1}, v_{ℓ_2}).

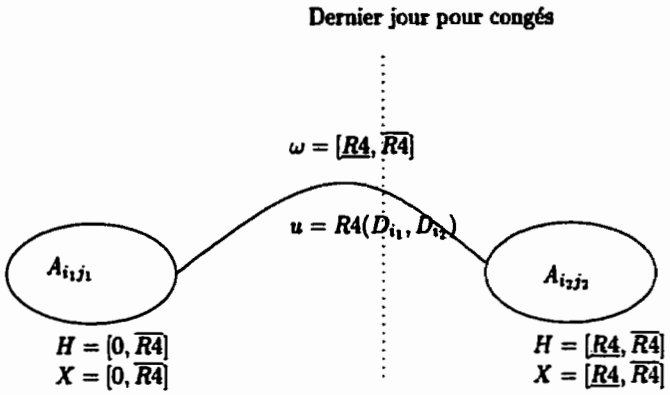


Figure 2.16: Cas noeud cible après dernier jour candidat pour congés (ressource $R4$)

Cas des arcs partant de la source

La source se situe forcément avant le premier jour à accorder congés. On aura donc les données suivantes sur les arcs quittant la source :

$$\omega = [0, \overline{R4} - R4(D_{i_1}, D_{i_2})]; u = R4(D_{i_1}, D_{i_2}).$$

Cas des arcs arrivant au puits

De la même manière, les arcs menant au puits porteront les données suivantes :

$$\omega = [\underline{R4} - R4(D_{i_1}, D_{i_2}), \overline{R4} - R4(D_{i_1}, D_{i_2})]; u = R4(D_{i_1}, D_{i_2}).$$

Récapitulatif

- Cas où le noeud cible porte un jour qui se situe avant (ou qui est) le dernier jour où l'on peut accorder un congé :

$$\omega = [0, \overline{R4} - R4(D_{i_1}, D_{i_2})]; u = R4(D_{i_1}, D_{i_2}).$$

- Cas où le noeud cible porte un jour qui se situe après le dernier jour où l'on peut accorder un congé :

$$\omega = [\underline{R4} - R4(D_{i_1}, D_{i_2}), \overline{R4} - R4(D_{i_1}, D_{i_2})]; u = R4(D_{i_1}, D_{i_2}).$$

- Cas d'un arc passant par un noeud virtuel :

$$1^{\text{ère}} \text{ moitié} : \omega = [0, \overline{R4}]; u = 0.$$

$$2^{\text{ème}} \text{ moitié} : \text{données présentes sur l'arc } (v_{\ell_1}, v_{\ell_2}) \text{ sans noeud fictif.}$$

- Cas des arcs quittant la source :

$$\omega = [0, \overline{R4} - R4(D_{i_1}, D_{i_2})]; u = R4(D_{i_1}, D_{i_2}).$$

- Cas des arcs arrivant au puits :

$$\omega = [\underline{R4} - R4(D_{i_1}, D_{i_2}), \overline{R4} - R4(D_{i_1}, D_{i_2})]; u = R4(D_{i_1}, D_{i_2}).$$

2.7 Note de conclusion

Le lecteur pourra remarquer que la modélisation proposée ici, tout comme celle de Semet, Jaumard et Vovor ([1], [2]), fait intervenir pour les fenêtres de mises à jour sur les arcs deux intervalles notés X et H. Ceci est hérité de l'algorithme de Vovor, qui a servi de base à notre travail.

Cependant la modélisation proposée ici est totalement nouvelle, et prouve que l'on peut trouver une modélisation en n'ayant plus qu'un seul intervalle de mise à jour sur les arcs. Cette modélisation a été complètement repensée par rapport à celle de Vovor [2]. Même si elle se base sur les mêmes définitions logiques et sémantiques des ressources, la formulation proposée ici est complètement novatrice. En effet, le lecteur aura remarqué que pour tous les noeuds du graphe, on peut trouver une modélisation dans laquelle $H = X$ pour chacune des ressources. Comme on l'a vu dans la section 2.2.4, cette réalisation permet de rendre le modèle théoriquement compatible avec l'application Gencol [12] (voir section 2.2.4).

Il est pourtant utile de garder au sein du programme la distinction entre les intervalles X et H. En effet, nous avons réussi, dans le cas présent, à unifier ces deux intervalles pour les quatre ressources prises en compte. Mais rien

n'indique que ceci est possible pour toutes les ressources que l'on serait amené à prendre en compte. En effet, on pourrait imaginer une ressource pour laquelle un dépassement des valeurs permises sur les noeuds (donc une valeur courante de la ressource sortant de l'intervalle H) devrait se traduire par un traitement particulier, donc par une valeur particulière de mise à jour (sur l'intervalle X). Cette valeur particulière permettrait alors d'emprunter un arc réservé au traitement de cette valeur. Le logiciel développé étant en voie d'intégration dans des unités telles que les blocs opératoires et les urgences, il n'est pas impossible de voir apparaître des contraintes qui nécessiteraient d'utiliser deux intervalles distincts.

Chapitre 3

Modifications et améliorations

Dans ce chapitre, nous présentons diverses modifications qui ont été apportées au programme. Dans un premier temps, nous parlons de la distinction que nous avons éclaircie entre les jours de congés (non travaillés mais payés) et les jours de repos (non travaillés et non payés).

Nous montrons ensuite comment l'implémentation d'une fonction d'arrondi nous permet d'accélérer la sortie de solutions réalisables, permettant ainsi d'avoir rapidement une idée de la qualité des horaires produits.

Ensuite, une section est consacrée aux améliorations théoriques apportées mais non encore implantées dans le programme. Nous terminons sur la présentation d'un développement théorique dont l'implantation au sein de l'outil logiciel a été annulée, et nous montrons les raisons de cette annulation.

3.1 Distinction congés - repos

Une des premières améliorations envisagées pour le programme IRIS était de pouvoir effectuer la distinction entre les différents jours de non-travail. Par définition, dans la suite de ce chapitre, on parlera de jour de congé pour une journée non travaillée mais payée, et de jour de repos pour une journée non travaillée et non

payée. Les deux types de journées sont dites journées non travaillées.

Dans un premier temps, nous examinerons les raisons qui nous ont poussés à chercher une nouvelle modélisation pour la ressource qui gère les jours de congés. Nous examinerons ensuite la nouvelle modélisation que nous avons retenue pour la ressource $R4$. Nous terminerons cette section en exhibant les différences que cette nouvelle modélisation entraîne pour les autres ressources.

Le lecteur pourra noter qu'il existe un troisième type de jours non travaillés, qui sont les congés sans solde. Pour l'instant, ces congés ne sont pas pris en compte par le modèle, et les journées ou périodes étant accordées comme congés sans solde à des infirmières seront marquées comme telles par les infirmières responsables avant de lancer l'optimisation dans le logiciel développé.

3.1.1 Motivations pour la nouvelle modélisation

Dans la première version du programme de génération d'horaires, la différence entre les deux types de jours non travaillés était faite à l'interne pour comptabiliser le respect du nombre de jours de congés à accorder, mais l'information était perdue par la suite. En effet, on contrôlait le respect du nombre de jours de congé, mais on n'assignait pas ces jours de congé à des dates précises.

Plusieurs raisons nous ont alors poussés à chercher une nouvelle méthode pour modéliser les journées de congé à accorder à une infirmière dans l'horizon courant.

La première provient d'une remarque des hôpitaux partenaires. Le but de ce projet de génération automatisée d'horaires d'infirmières est d'intégrer à terme cette application dans leur module de gestion. Or, celui-ci a besoin de faire la différence entre jours de repos et jours de congé. Rappelons que les jours de congé sont payés, les jours de repos ne le sont pas. Il est donc nécessaire que l'outil de génération d'horaires fasse la différence entre les deux types de jours non travaillés pour pouvoir être couplé au système de paye.

La seconde raison provient de remarques faites pendant le développement de l'outil et surtout pendant la phase de validation théorique. On s'aperçoit en relisant le chapitre de validation théorique (voir chapitre 2) que la modélisation de la contrainte des jours de congés (ressource $R4$) a des conséquences sur les autres contraintes que les hôpitaux doivent prendre en compte lorsqu'ils nous transmettent leurs contraintes. Par exemple, si pendant une sous-période de l'horizon (au sens de la ressource $R1$) l'hôpital souhaite accorder des congés à une infirmière, il faut déduire le nombre d'heures de congés à accorder du minimum d'heures à travailler pour cette sous-période. Ceci est peu pratique, d'autant plus que les infirmières ont toutes des requêtes différentes en termes de congés, ce qui complique encore l'écriture du fichier d'entrée.

Il était donc nécessaire de trouver une meilleure modélisation pour gérer les jours de congés, modélisation originale qui est décrite dans la section 3.1.2.

3.1.2 Nouvelle modélisation pour les jours de congés

Dans cette section, nous allons reprendre les notations introduites dans la section 2.6.1. Nous rappelons ici à des fins de clarté les principales données dont nous disposons pour gérer la ressource reliée à l'allocation des jours de congés. Nous avons donc à notre disposition :

- les jours extrêmes entre lesquels l'infirmière est autorisée à prendre ses congés;
- les nombres minimaux et maximaux de jours de congés que l'infirmière est autorisée à prendre pendant l'horizon courant. On notera $\underline{R4}$ et $\overline{R4}$ ces bornes;
- une liste de jours candidats que l'infirmière peut prendre congé. Si cette liste est vide, on considère que tous les jours situés entre le jour au plus tôt et le jour au plus tard sont candidats pour être des congés.

Le principe de la nouvelle modélisation retenue pour la ressource $R4$ est le suivant : des noeuds correspondant à des couples (jour D_i, quart de congé) vont être ajoutés dans le graphe de l'infirmière considérée. Un couple va être ajouté pour chaque jour de l'horizon appartenant à la liste des jours candidats à être congés. Le principe sera alors pour la ressource de faire le compte du nombre de tels sommets rencontrés lors du parcours du graphe. Pour être réalisable, un chemin devra contenir un nombre de tels sommets compris entre $\underline{R4}$ et $\overline{R4}$.

Comme dans le chapitre 2, la démarche pour expliquer la nouvelle modélisation va être classique : on définira d'abord les étiquettes sur les noeuds, puis les étiquettes portées par les arcs.

3.1.2.1 Étude des étiquettes sur les noeuds

Pour cette ressource, les noeuds du graphe peuvent être divisés en trois catégories : les noeuds correspondant à un jour accordé en congés, les noeuds correspondant à des jours non congés et les noeuds dits spéciaux (source, puits et noeuds virtuels).

- Pour les noeuds correspondant à des jours de congés, la ressource peut être comprise entre 1 et $\overline{R4}$. On aura donc $H = X = [1, \overline{R4}]$.
- Pour les noeuds non congés, on aura de manière évidente $H = X = [0, \overline{R4}]$.
- Sur la source, on doit contrôler que la ressource initiale est bien comprise entre les bornes extrêmes. On aura donc : $H = X = [0, \overline{R4}]$. Pour les noeuds virtuels du graphe, on ne doit pas contrôler une autre ressource que $R2$. On doit donc, pour $R4$, avoir des valeurs permissives telles que : $H = X = [0, \overline{R4}]$. Pour le puits, on doit avoir effectué un nombre de jours de congés compatibles avec les données du problème. On aura donc : $H = X = [\underline{R4}, \overline{R4}]$.

Nous allons donc maintenant voir quelle est l'étiquette que doit porter chaque arc du graphe pour conduire à un chemin réalisable.

3.1.2.2 Étude des étiquettes sur les arcs

Nous allons étudier les fenêtres de ressources à placer sur les arcs du graphe pour la ressource $R4$. Nous commencerons par les arcs dits classiques pour finir par les arcs particuliers. Dans le premier cas, nous diviserons les noeuds en deux catégories : les noeuds portant un quart effectivement travaillé et les noeuds portant un quart de congé.

Pour étudier les fenêtres de ressources à placer sur les arcs dits réguliers, on peut distinguer quatre cas distincts (voir figure 3.1) :

- Pour un arc partant d'un noeud travaillé et arrivant à un autre noeud travaillé, il faut autoriser toutes les valeurs réalisables de la ressource. La consommation sera bien évidemment égale à 0. On aura donc sur cet arc :

$$\omega = [0, \overline{R4}]; u = 0.$$

- Pour un arc partant d'un noeud travaillé et arrivant à un noeud de congé, la ressource $R4$ doit avoir une valeur comprise entre 0 et $\overline{R4} - 1$, puisqu'on se prépare à accorder un jour de congé. La consommation sera égale à 1. On aura donc :

$$\omega = [0, \overline{R4} - 1]; u = 1.$$

- Pour un arc partant d'un jour de congé pour arriver à un quart travaillé, la ressource doit avoir une valeur comprise entre 1 (puisqu'on vient d'accorder une journée de congés), et $\overline{R4}$. La consommation sera égale à 0. On aura donc :

$$\omega = [1, \overline{R4}]; u = 0.$$

- Pour un arc quittant un jour de congé et arrivant à un autre jour de congé, la ressource aura une valeur comprise entre 1 et $\overline{R4} - 1$. La consommation sera égale à 1. On aura donc sur cet arc :

$$\omega = [1, \overline{R4} - 1]; u = 1.$$

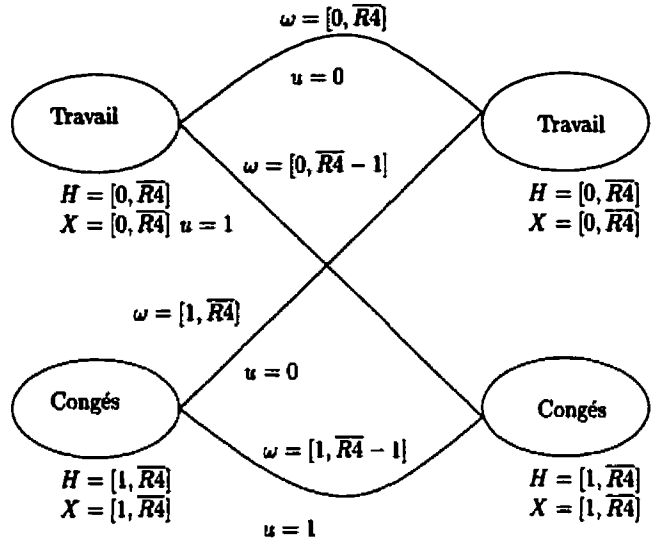


Figure 3.1: Nouvelle modélisation (ressource $R4$)

Pour les arcs dits particuliers (partant de la source, arrivant au puits ou passant par un noeud virtuel), les cas de figure sont décrits ci-dessous.

- Pour les arcs partant de la source, on doit autoriser toutes les valeurs réalisables de la ressource car les jours de congés peuvent dans certains cas être gérés sur deux horizons consécutifs. On aura alors :

$$\omega = [0, \overline{R4}]; u = 0$$

pour les arcs dont le noeud cible représente un quart de travail, et

$$\omega = [0, \overline{R4} - 1]; u = 1$$

pour les arcs dont le noeud cible représente un jour de congés.

- Pour les arcs arrivant au puits, il suffit de contrôler que la valeur de la ressource est bien comprise entre les bornes de réalisabilité. La consommation sera bien entendue de 0. On aura donc sur les arcs arrivant au puits :

$$\omega = [\underline{R4}, \overline{R4}]; u = 0.$$

- Pour le cas d'un noeud fictif présent sur un arc (v_{ℓ_1}, v_{ℓ_2}), la situation est simple : le noeud fictif ne servant que pour la ressource gérant les fins de semaine, un formalisme est adopté. Sur la première moitié de l'arc, on fait figurer un test non contraignant. Sur la deuxième moitié, on fait figurer les informations qui auraient normalement été présentes sur l'arc (v_{ℓ_1}, v_{ℓ_2}). On retrouve donc les étiquettes suivantes :

$$1^{\text{ère}} \text{ moitié} : \omega = [0, \overline{R4}]; u = 0;$$

$2^{\text{ème}}$ moitié : données présentes sur l'arc(v_{ℓ_1}, v_{ℓ_2}) sans noeud fictif.

3.1.3 Répercussions sur les autres ressources et sur la construction du graphe

Nous présentons dans cette section les répercussions de la modification de la ressource gérant les jours de congés, que ce soit du point de vue de la construction des graphes que de celui de la gestion des trois autres ressources prises en compte dans le problème.

3.1.3.1 Modifications dans la construction du graphe

Comme on l'a vu dans la section précédente, la principale modification qui intervient dans la construction du graphe est l'ajout de sommets qui correspondent à une paire (jour candidat pour un congé D_i, "quart" congé). On ajoutera donc autant de noeuds dans le graphe qu'il y a de jours candidats à être congés.

L'ajout de ces nouveaux sommets aura également une conséquence au niveau des arcs : alors qu'avec l'ancienne modélisation, il fallait ajouter le nombre de jours candidats à être congés au nombre maximum de jours de repos consécutifs à accorder pour avoir la distance maximale (en terme de jours) entre deux sommets, avec la nouvelle modélisation, cette distance est inférieure ou égale au nombre maximum de jours consécutifs de repos que l'on peut accorder.

Note : Le lecteur remarquera (voir explication à la section 3.1.3.3) que l'on ne peut affecter un jour de congé un jour de fin de semaine.

3.1.3.2 Modifications pour la ressource "charge de travail"

Comme décrit dans les motivations qui nous ont poussés à adopter cette nouvelle modélisation pour les jours de congés, une des raisons du changement est la possibilité dans le futur de coupler notre outil de génération d'horaires à un système de gestion des ressources humaines. Il est donc nécessaire d'associer à chaque quart de congé un nombre d'heures qui correspondra au nombre d'heures payées pour une journée de congés.

Fort heureusement, cette durée est standard pour toutes les unités au sein d'un même hôpital. On va donc associer à chaque noeud représentant une journée de congés une durée correspondant au nombre d'heures payées. A partir de ce point, il est possible de traiter ces noeuds de congé comme tous les autres noeuds du graphe en ce qui concerne la ressource assurant le suivi de la charge de travail. On se reportera donc on chapitre 2 pour la gestion de cette ressource.

Note : Il n'est plus nécessaire de déduire le nombre de jours de congés du minimum de la charge de travail à respecter comme on le faisait précédemment, puisque les jours de congés ont une durée qui entre en compte dans la comptabilisation de la charge de travail.

3.1.3.3 Modifications pour la ressource "fins de semaine"

Comme on le verra dans la section 3.3, certaines contraintes ne sont pas encore clairement définies en ce qui concerne la relation fin de semaine - jours de congés. Cependant, certains points étaient clairs et ont servi de base à la modification de la ressource "fins de semaine" pour prendre en compte les jours de congés. Ces points certains sont définis dans les différentes conventions collectives [17] et sont les suivants pour l'hôpital Royal Victoria :

- On ne peut affecter un jour de congé un samedi ou un dimanche.
- Une infirmière, si elle doit travailler la fin de semaine avant une semaine de congé, ne travaillera que le samedi de la fin de semaine.
- Au retour d'un congé, une infirmière ne peut travailler le samedi de la fin de semaine du retour. Si nécessaire dans la rotation fin de semaines travaillées / non travaillées, l'infirmière travaillera le dimanche de son retour.

Étant donné que cette modélisation n'est pas appliquée par tous les hôpitaux et que même pour l'hôpital avec lequel nous avons travaillé sur le sujet les différents points semblent flous, il a été choisi d'adopter le formalisme suivant quant à la gestion de la ressource des fins de semaine pour les jours de congés :

- Pour la construction du graphe, on respecte la règle qui stipule qu'on ne peut affecter un quart de congé un jour de fin de semaine.
- Pour la gestion de la ressource proprement dite, la solution retenue est de remettre à zéro la valeur de la ressource $R2$ à chaque passage sur un jour de congé. En effet, remettre à zéro la valeur de la ressource est la solution qui permet le plus de flexibilité dans la gestion des fins de semaine. Sur un noeud représentant un jour de congé, on aura donc pour la ressource $R2$: $H = X = [0, 0]$.

3.1.3.4 Modifications pour la ressource "rotation des types de quart"

Les modifications induites par la nouvelle modélisation de la ressource gérant les jours de congés sur les règles de rotation des types de quart (ressource notée $R3$ dans la chapitre 2) sont très simples. En effet, un jour de congé au milieu d'une série de quarts d'un certain type doit réinitialiser la valeur courante de la ressource gérant les types de quart. Il suffit donc de poser, pour chaque noeud associé à un jour de congé, pour la ressource $R3$: $H = X = [0, 0]$. Ainsi, on force la réinitialisation de la valeur courante de la ressource.

3.2 Fonction d'arrondi

Une amélioration a été apportée à la fin de la phase de développement de IRIS, et a été inspirée par les résultats que nous obtenions sur les tests fournis par l'hôpital Royal Victoria. En effet, les personnes ressources désirent obtenir relativement vite un premier horaire réalisable, dans le but d'évaluer la qualité des solutions subséquentes et de voir s'il est utile de laisser tourner le logiciel jusqu'à l'optimalité.

Dans cet objectif, une procédure a été développée pour tenter, à chaque itération du simplexe dans le problème maître, d'extrapoler une solution réalisable. Dans l'explication qui suit, on reprend les notations du chapitre 1.

3.2.1 Problème rencontré

Une solution est dite réalisable pour le problème maître si elle satisfait toutes les contraintes de ce problème. Pour peu de disposer d'une fenêtre de réalisabilité suffisamment étendue, les contraintes de quotas ne sont pas les plus difficiles à satisfaire. La difficulté provient de la contrainte d'intégralité des variables de décisions y_{ks} associées aux horaires. Cette contrainte se formule de la façon suivante :

$$y_{ks} \in \{0,1\}, \forall k, \forall s.$$

Cependant, il faut se rappeler qu'à chaque noeud de l'arbre de branchement, cette contrainte est relaxée. On va donc avoir plusieurs horaires pour une infirmière donnée, chacun avec une variable de décision y_{ks} associée ayant une valeur inférieure à un. Le branchement se charge de rétablir l'intégralité de cette variable, mais il pourrait être intéressant de disposer d'une procédure permettant, à partir d'une solution fractionnaire, d'extrapoler une solution entière réalisable.

3.2.2 Solution retenue

Une méthode simple pour tenter d'extrapoler une solution entière réalisable a été implémentée. Elle permet, dans certains cas, de produire une première solution réalisable dès le noeud racine de l'arbre de branchement. Ceci permet à l'utilisateur du logiciel de disposer d'une première version de son horaire dans des temps tout à fait raisonnables (voir chapitre 4).

La non réalisabilité la plus fréquemment rencontrée provenant de la non intégralité des variables de décisions y_{ks}, la méthode suivante est appliquée. A chaque itération du simplexe dans le problème maître, on applique l'algorithme suivant :

- Pour toute infirmière N_k, chercher $s_0 = argmax_s(y_{ks})$
- Vérifier si les horaires H_{ks_0} définissent un horaire réalisable.

Il est ainsi possible d'obtenir des horaires réalisables beaucoup plus rapidement qu'en laissant tourner l'outil sans cette modification. Ceci est particulièrement utile lorsque utilisé en conjonction avec l'option qui permet de sortir du programme dès que l'on dispose d'un horaire réalisable.

3.3 Contraintes à rajouter

La première phase de développement et de test s'est achevée fin septembre 2000. A cette date, l'outil développé, IRIS, incluait toutes les contraintes exprimées par les hôpitaux lors de la définition du projet dans la première moitié de 1999. Cependant, au fil des tests et des itérations entre l'université et les hôpitaux, on s'est aperçu que certaines contraintes n'avaient pas été exprimées lors de la définition du problème. Ces contraintes ne sont donc pas prises en compte dans la présente version du programme mais devraient être intégrées dans une version future. L'objet de cette section est donc de passer en revue les différentes contraintes qui seront ajoutées lors des phases de développement subséquentes à notre travail.

Nous commencerons par exprimer la contrainte qui entraînera probablement les modifications les plus importantes au sein du programme, puis nous passerons en revue les quelques contraintes mineures à rajouter qui ne feront intervenir que la modification de certains arcs ou sommets du graphe.

3.3.1 Contrainte des jours de travail consécutifs

La première contrainte à ajouter à la définition du problème est probablement celle qui demandera le plus de modifications dans la modélisation du problème. Cette étude est ici fournie pour permettre au lecteur de comprendre le cadre de ces modifications.

En examinant les premiers fichiers de résultats, les deux organismes partenaires du projet, soit le CHUM informatique et l'hôpital Royal Victoria, nous ont fait part du non respect d'une contrainte qui est pour eux essentielle : le respect d'une durée maximale (en termes de nombres de jours consécutifs) pour une période travaillée. Il faut savoir que cette durée maximale est de 5 jours pour les hôpitaux du CHUM et de 7 jours pour l'hôpital Royal Victoria. Ces durées maximales sont définies dans les différentes conventions collectives [17].

La modélisation envisagée, qui devrait être implantée dans une prochaine version, serait de faire intervenir une ressource supplémentaire qui permettrait de "mesurer" la longueur (en munissant notre espace d'une distance consistant à compter le nombre de jours consécutifs entre deux dates) d'une série de jours consécutifs travaillés. Nous allons donc passer en revue les grandes lignes de cette ressource, notamment son implémentation en termes de fenêtres sur les noeuds et les arcs.

Dans la suite de cette section, nous nommerons $R5$ cette ressource. On notera également $\overline{R5}$ le nombre maximum de jours consécutifs travaillés.

3.3.1.1 Fenêtres sur les sommets

Le principe de cette ressource supplémentaire est relativement simple : il s'agit de compter le nombre de jours consécutifs travaillés et de vérifier qu'en tout temps, la valeur de la ressource est bien comprise entre 0 et le nombre maximum de jours consécutifs travaillés, $\overline{R5}$.

Sur les noeuds représentant des jours travaillés, il faut contrôler que la ressource a bien une valeur comprise entre 1 et $\overline{R5}$. Si tel n'est pas le cas, on remettra la valeur à jour avec l'une ou l'autre borne suivant le sens du dépassement. On aura donc sur tous les noeuds correspondants à des jours travaillés $H = X = [1, \overline{R5}]$.

Sur les noeuds de congés et les noeuds particuliers (source si elle est virtuelle, puits et autres noeuds virtuels), il faut remettre la ressource à zéro. Sur de tels noeuds, on portera les valeurs $H = X = [0, 0]$.

3.3.1.2 Fenêtres sur les arcs

Sur les arcs, on doit contrôler deux choses : si on est dans une série travaillée, la valeur de la ressource doit être comprise entre 0 et $\overline{R5}$, et on incrémente cette valeur de 1 à chaque sommet travaillé rencontré. Si on arrive sur un sommet de congé ou bien si on utilise un arc qui franchit plusieurs jours de repos, on doit contrôler qu'à la sortie, la valeur de la ressource était bien inférieure à $\overline{R5}$ et remettre la valeur courante de la ressource à 0. La mise à zéro sera effectué par les fenêtres de contrôle sur les noeuds.

Dans la suite de la section, on notera $d(D_{i_1}, D_{i_2})$ la distance (en termes de nombre de jours) entre deux jours D_{i_1} et D_{i_2}.

- Si $d(D_{i_1}, D_{i_2}) = 0$, et si le quart porté par le sommet cible est un quart de travail, alors on aura $\omega = [0, \overline{R5} - 1]; u = 1$.
- Si $d(D_{i_1}, D_{i_2}) = 0$, et si le quart porté par le sommet cible est un quart de congés ou un si le sommet cible est virtuel, on aura $\omega = [0, \overline{R5}]; u = -\overline{R5}$.

- Si $d(D_{i_1}, D_{i_2}) > 0$, et si le quart porté par le sommet cible est un quart de travail, alors il faut vérifier que la séquence de travail (si on sort d'une séquence de travail) avait la bonne longueur et réinitialiser la nouvelle. On aura donc : $\omega = [0, \overline{R5}]; u = -\overline{R5}$.
- Si $d(D_{i_1}, D_{i_2}) > 0$, et si le quart porté par le sommet cible est un quart de congés ou un si le sommet cible est virtuel, on aura $\omega = [0, \overline{R5}]; u = -\overline{R5}$.

3.3.2 Autres contraintes à ajouter

Dans cette section, nous listons toutes les contraintes qui nous ont été transmises lors de l'examen des premiers résultats. Ces contraintes ne sont pas incluses dans la version initiale du logiciel IRIS, mais devront être incorporées dans une version ultérieure du programme.

Le lecteur pourra noter que ces contraintes proviennent de remarques faites par les intervenants de l'hôpital Royal Victoria, et ne s'appliquent peut-être pas de la même manière dans d'autres centres hospitaliers. Ces contraintes sont définies pour la plupart dans la convention collective de l'hôpital Royal Victoria [17].

- Une fin de semaine ne doit jamais être coupée : si le samedi est travaillé, le dimanche doit l'être aussi et inversement si le samedi est chômé, le dimanche le sera aussi.

  La seule exception à cette règle intervient dans le cas d'une fin de semaine précédant une semaine de congé. Dans ce cas, si la fin de semaine doit être travaillée (au regard de la règle de rotations des fins de semaine), alors seul le samedi sera travaillé, la semaine de congé commençant le samedi à minuit. De même, au retour d'une période de congé, si l'infirmière doit travailler la fin de semaine de son retour (respect des règles de rotation), alors elle ne travaillera que le dimanche et la fin de semaine comptera comme travaillée.
- Lors de périodes de congé, le calcul de la rotation entre fins de semaines

travaillées et non travaillées continue de manière normale même si tous les jours de fins de semaine sont congés. C'est-à-dire que si une infirmière doit travailler une fin de semaine entre deux semaines de congés pour respecter sa rotation de fins de semaine, alors la fin de semaine sera comptée comme travaillée mais sera chômée par l'infirmière.

Ces deux nouvelles contraintes sont rendues complexes à implémenter du fait de la versatilité que l'on veut conserver pour l'outil développé. En effet, certaines unités vont appliquer ces règles en l'état, tandis que d'autres utilisent des variantes (par exemple, dans certaines unités les infirmières ne travaillent pas le dimanche). Il faudra donc songer à une modélisation à l'aide de noeuds virtuels pour couvrir tous les cas possibles. Cette modélisation fait partie des travaux futurs à réaliser.

3.4 Profils d'infirmières

Une des difficultés rencontrées lors du développement et du test du logiciel était l'occupation mémoire très importante du programme, une fois lancé. En effet, outre la construction d'un graphe (problème auxiliaire) pour chaque infirmière présente dans l'unité à gérer, il faut aussi stocker pour chaque graphe la liste des vecteurs ressources réalisables en chacun des noeuds, dans le but d'aboutir à un algorithme de même complexité qu'un plus courts chemins dans un graphe acyclique classique [2].

Nous avons donc imaginé plusieurs scénarios pour tenter de contourner ce problème d'occupation mémoire. En particulier, une des idées récurrentes que nous voulions tester était de regrouper des infirmières ayant des contraintes similaires au sein de profils d'infirmières, chacun de ces profils ayant un graphe générique (d'où la réduction de l'occupation mémoire par la diminution du nombre de graphes présents pour les problèmes auxiliaires).

Nous allons donc dans cette section étudier l'approche suivie pour la mo-

délisation de ces profils et nous montrerons que leur mise en place ne se serait finalement pas révélée avantageuse en ce qui concerne l'occupation mémoire de l'outil développé.

3.4.1 Modélisation des profils

Le but est donc de regrouper plusieurs infirmières au sein d'un seul "profil" et de ne construire qu'un seul graphe pour toutes les infirmières présentes dans le profil considéré. L'objectif était donc, pour avoir le meilleur gain en termes d'espace mémoire occupé, de regrouper le plus grand nombre possible d'infirmières au sein d'un même profil.

Pour comprendre la démarche que nous envisagions, rappelons qu'un horaire réalisable pour une infirmière doit satisfaire plusieurs contraintes. Il doit en particulier satisfaire six contraintes "dures" (affectations interdites et obligatoires, charge de travail, rotation des fins de semaine, rotation entre les différents types de quarts et respect des jours de congés). C'est sur ces critères que nous envisagions de faire le partitionnement des infirmières.

Cependant, pour être réalistes et à la vue des premières expériences menées avec les hôpitaux partenaires, nous savions que nous n'avions que peu de chances de trouver ne serait ce que deux infirmières avec des contraintes dures identiques. L'idée était donc de trouver les contraintes sur lesquelles les infirmières n'avaient que peu de différences, de garder ces contraintes pour la construction du graphe comme des contraintes dures, mais de relaxer les autres contraintes, par exemple, à l'aide d'une méthode de pénalisation.

Le critère de choix des contraintes dures qui serviraient au partitionnement était basé sur l'expérience des premiers fichiers de tests que nous avions reçus. Les contraintes choisies étaient les suivantes :

- La charge de travail : en effet, la plupart des infirmières travaillent à temps plein. Pour celles travaillant à temps partiel, les différentes catégories de

personnel sont bien déterminées et la charge de travail est clairement définie (par exemple, on peut avoir une infirmière qui fera 30% d'une infirmière travaillant à temps plein).

- Les rotations de fins de semaines : la plupart des infirmières obéissent à la rotation stricte une fin de semaine sur deux travaillée.

Bien évidement, pour regrouper deux infirmières au sein d'un même profil, il faut qu'elles aient le même graphe. Il faut donc qu'elles aient les mêmes affectation interdites, puisque ces affectations sont modélisées en enlevant le sommet correspondant dans le graphe.

Nous voyons donc quels sont les critères qui avaient été choisis pour effectuer les regroupements d'infirmières. Les autres contraintes dures (rotations des types de quart et jours de congés) sont trop peu semblables d'une infirmière à l'autre pour espérer les faire entrer dans le critère de partitionnement. Nous allons d'ailleurs voir dans la prochaine section pourquoi ces deux contraintes ont contribué à nous faire abandonner le projet de regroupement d'infirmières en profils.

3.4.2 Raisons de l'abandon de l'idée des profils d'infirmières

L'idée de regrouper les infirmières dans des profils a été rapidement abandonnée, et ceci pour deux raisons principales que nous allons décrire dans les paragraphes suivants.

3.4.2.1 Impossibilité de créer des profils regroupant plus que quelques infirmières

Comme on l'a vu, les critères de choix pour construire la partition sur l'ensemble des infirmières sont le respect des mêmes affectations interdites, et la même charge

de travail, ainsi que les mêmes règles de rotation de fins de semaine.

A priori, ces contraintes semblent être suffisamment standards pour permettre de regrouper les infirmières en profils regroupant plus de deux ou trois personnes. Pourtant, l'expérience prouvera que ceci n'est pas forcément vrai, puisque très souvent des particularités viennent s'insérer dans les cas standards (on citera par exemple le cas d'une infirmière qui part en formation, et dont la charge de travail sera diminuée en conséquence, alors que l'infirmière travaille dans une catégorie standard de temps plein).

3.4.2.2 Espace mémoire gagné négligeable

Comme on l'a vu plus haut, on ne peut pas utiliser toutes les contraintes dures comme critères de partitionnement lors de la construction des profils, car certaines contraintes varient trop d'une infirmière à l'autre. Il nous reste notamment les contraintes de rotations de types de quart et les contraintes de jours de congés à accorder.

L'idée est donc de créer un graphe par profil d'infirmières résultant du partitionnement, puis de créer sur ce graphe des pointeurs qui permettront de modéliser les différences entre chaque infirmière d'un même profil. Autrement dit, il faut créer le graphe, puis en lieu et place des fenêtres de ressources sur les noeuds et les arcs, mettre des pointeurs vers des fenêtres de ressources, qui, elles, dépendront de l'infirmière considérée lors de l'appel au problème auxiliaire.

En fait, l'espace mémoire gagné est négligeable. En effet, on ne stocke plus qu'un seul graphe par profil d'infirmières. Certes, si on admet qu'en moyenne on peut regrouper deux ou trois infirmières par profil, on divise l'espace occupé par les graphes par deux ou trois suivant le cas. Cependant, comme on l'a vu plus haut, toutes les contraintes ne sont pas utilisées pour générer les profils d'infirmières. Il reste au moins donc deux ressources qui dépendent encore de chaque infirmière. Il faut donc stocker pour chaque infirmière un ensemble de vecteur de ressources

ayant la même structure que le graphe de départ puisqu'on doit avoir, pour chaque sommet, la liste des vecteurs de ressources qui conduisent à un chemin réalisable.

Pour résumer la situation, en construisant des profils d'infirmières, on s'aperçoit que l'on est encore obligé de stocker de manière indépendante pour chaque infirmière des informations ayant la même structure que celle d'un graphe. L'espace mémoire gagné est donc négligeable.

Pour toutes ces raisons, il a donc été décidé d'abandonner le projet de regrouper les infirmières en profil.

Chapitre 4

Résultats de calcul

4.1 Introduction

Ce chapitre final sert à présenter les différents résultats obtenus tout au long du projet de recherche mené à bien pendant la maîtrise. Il se veut un regard critique sur les résultats obtenus versus ceux escomptés. Il est donc divisé en plusieurs sections.

Dans la première section, nous donnons un bref aperçu des ressources informatiques utilisées lors du développement du programme.

Dans une deuxième section, nous présentons les données dont nous disposons pour effectuer les tests sur notre programme et par là même sur le modèle mathématique associé. Les fichiers d'entrée fournis pour effectuer les tests étant longs et complexes, ils ne seront pas fournis mais plutôt décrits au lecteur. Seul un fichier de test artificiel sera donné en annexes pour que le lecteur puisse voir le format des entrées utilisées.

Nous présentons ensuite les résultats de calcul à proprement parler : les horaires qui ont été générés par le programme IRIS pour les hôpitaux partenaires, en comparaison avec ceux confectionnés pour les mêmes périodes de manière ad-hoc par les infirmières chefs. Un commentaire critique sera alors apporté.

4.2 Ressources informatiques utilisées pour le développement

Nous présentons de manière succincte dans cette section les différents outils logiciels que nous avons utilisés pour développer le logiciel IRIS. Nous commençons par présenter les outils reliés au développement proprement dit, puis nous présentons les grandes lignes des deux outils d'optimisation que nous utilisons.

4.2.1 Outils reliés au développement du logiciel

Le développement du logiciel IRIS a été effectué sous plate-forme Windows NT 4.0. Les hopitaux partenaires travaillaient en effet sous cet environnement plutôt que sous Unix. Le choix de l'environnement de développement s'est donc effectué en fonction de ce système d'exploitation. Le choix s'est donc porté sous l'environnement de développement de Microsoft, Visual Studio.

Le programme étant développé dans le langage C++, le compilateur utilisé était celui fourni par Microsoft dans sa suite de développement Visual C++. Nous avons utilisé la version 6.0 de cet environnement.

Pour le déverminage du programme, nous avons utilisé le débugueur standard de Visual C++. Pour parfaire la mise au point du programme, nous avons utilisé l'outil Purify de Rational Software, dans sa version 5.0. Cet outil nous a permis de tracer les bugs les plus sérieux présents dans le programme et également de nettoyer la plupart des fuites de mémoire présentes dans les premières versions développées.

Ceci résume en quelques lignes les outils que nous avons utilisés pour le développement. Nous allons maintenant voir sur quels outils nous nous sommes appuyés pour notre algorithme d'optimisation.

4.2.2 Outils d'optimisation utilisés

Pour la partie optimisation, nous avons choisi de baser notre modélisation et donc notre outil sur deux logiciels bien connus dans le monde de l'optimisation et dont l'efficacité n'est plus à démontrer, même si quelques reproches peuvent être formulés à l'encontre du programme ABACUS.

Le premier outil auquel nous avons fait appel est un outil prenant en charge les algorithmes de séparation et évaluation progressive. Cet outil est en fait un ensemble de classes abstraites desquelles l'utilisateur fait dériver les classes maîtresses de son application. Cet outil, ABACUS ([18] et [19]), a été utilisé dans sa version 2.2. Cette version, bien que semblant très efficace en termes d'implantation et de qualité des algorithmes sous-jacents, présente l'inconvénient de requérir un espace mémoire très important pour l'exécution du programme. Néanmoins, cet outil propose toujours un excellent rapport facilité d'utilisation / désavantages, et son utilisation se justifie parfaitement.

Le second outil est utilisé de manière indirecte puisqu'il n'est appelé nulle part de manière explicite dans le programme. Il s'agit du logiciel Cplex version 6.0 [20]. Cet outil présente à la fois un programme interactif et un ensemble de classes (une API) pour résoudre des programmes linéaires en utilisant l'algorithme du simplexe. L'outil n'est pas utilisé directement dans notre code, puisque les appels se font à travers ABACUS. C'est en effet ABACUS qui se charge d'appeler les algorithmes de résolutions de Cplex lorsque nécessaire.

4.3 Présentation des données

Dans cette section, nous présentons les différents jeux de données que nous avons eu à notre disposition pour effectuer les tests sur le programme IRIS. Nous détaillons les fichiers tests obtenus à la fois du CHUM informatique et de l'hôpital Royal Victoria.

Le but de cette section est de permettre au lecteur intéressé de reproduire les tests le plus facilement possible. A cette fin, nous donnons quelques unes des caractéristiques de la machine sur laquelle nous avons fait tourné les tests.

- Processeur Intel Pentium III cadencé à 550 MHz.
- 256 Mo de mémoire vive, à laquelle nous ajoutons 128 Mo de mémoire virtuelle (cache sur le disque dur).
- Système d'exploitation Microsoft Windows NT 4.0 (service pack 5).
- Le programme testé est l'outil IRIS dans sa version 1.1 (archivée).

4.3.1 Données du CHUM informatique

Le CHUM est un regroupement d'hôpitaux dépendants de l'Université de Montréal. Le CHUM informatique est une association qui assure l'intégration et la mise à disposition de logiciels de gestion au sein de ces hôpitaux.

Les expériences menées en partenariat avec le CHUM informatique se sont faites sur un fichier créé de toutes pièces pour ces tests et ne représentant aucune situation réelle. Cependant, le fichier de test a eu le mérite d'arriver assez tôt durant la phase de développement du programme et a permis de tester les phases clefs du logiciel développé.

Une fois au point au niveau du formattage des données, le fichier de test présentait les caractéristiques suivantes :

- une période de planification de quatre semaines, soit vingt-huit jours pour l'horizon de planification;
- une équipe de sept infirmières qui devaient respecter certaines contraintes listées plus bas;
- trois quarts de travail réalisables (jour, soir et nuit), qui ont une durée de 7.25 heures chacun;
- des contraintes de quotas relativement uniformes. Chaque contrainte était

Nom du quart	Quota désiré	Déficit maximal	Surplus maximal
JOUR	3	1	1
SOIR	2	1	1
NUIT	2	1	1

Tableau 4.1: Contraintes de quotas (fichier du CHUM informatique)

donnée avec une fenêtre de réalisabilité de plus ou moins 1 sur la valeur cible. Ces contraintes étant les mêmes pour tous les jours de l'horizon, elles sont résumées dans le tableau 4.1.

Dans ce fichier, le personnel infirmier présentait les caractéristiques suivantes :

- toutes les infirmières travaillent à temps plein et doivent effectuer une charge de travail de 36.25 heures par semaine;
- quatre infirmières travaillent une fin de semaine sur deux, deux travaillent toutes les fins de semaine et pour la septième, aucune contrainte n'est exprimée sur les fins de semaine;
- une des infirmières peut effectuer n'importe quel enchaînement de types de quarts, les autres doivent avoir des séries de même quart ayant une longueur de 1 à 5, et des séries de congés de 2 à 4 jours;
- pour tester la ressource gérant les congés, une infirmière doit recevoir une journée de congé;
- diverses préférences et aversions sont exprimées pour tester le programme.

Le lecteur intéressé peut se reporter à l'annexe A pour voir un extrait du fichier d'entrée.

4.3.2 Données du centre des naissances (Hôpital Royal Victoria)

Nous allons dans cette section présenter les données ayant conduit aux résultats les plus significatifs obtenus durant cette période de recherche. Ces résultats ont été obtenus en partenariat avec l'hôpital Royal Victoria, et plus précisément avec l'unité "centre des naissances" (birthing center) de cet hôpital. Nous avons obtenu deux fichiers de test de cette unité correspondants à deux périodes de planification successives.

Cette unité a été choisie comme unité pilote pour les tests en accord avec les différents intervenants au sein du projet pour plusieurs raisons énumérées ci-bas.

- L'unité des naissances est une des plus grosses unités au sein de l'hôpital. Elle est constituée de 80 infirmières, auxquelles on ajoute les deux infirmières chefs qui se chargent de la planification des aspects ressources humaines au sein de l'unité.
- Cette unité est intéressante, car de nombreuses infirmières sont soit en congé de maternité, soit sur des absences diverses. Dans les faits, et sur la plupart des horizons de planification, sur les 80 infirmières appartenant à l'unité, seules 50 sont présentes. C'est le cas dans les deux fichiers de test reçu de cette unité.

On résume ici quelques-uns des grands traits de cette unité partenaire.

- On dispose de huit quarts de travail différents, répartis en trois types de quarts (jour, soir ou nuit). Ces quarts de travail ont des durées de douze, huit ou quatre heures. L'expérience montre que la plupart des infirmières n'ont qu'un ou deux quarts de travail réalisables parmi ces huit. L'infirmière ayant le plus grand nombre de quart réalisables peut effectuer quatre quarts de travail différents.
- Certains des quarts se recoupent ; on va donc définir une partition de la

journée en périodes de quotas. Les contraintes sont données en termes de périodes de quotas, mais ceci ne modifie en rien la modélisation du problème. On affecte toujours aux infirmières des quarts de travail, et on dispose en entrée d'une table qui spécifie quels quarts recouvrent quelles périodes. Le lecteur se reportera à la section 1.3.3 pour de plus amples informations. La figure 4.1 présente la partition de la journée retenue. Le lecteur remarquera que la partition proposée n'est pas de cardinalité minimale mais est conservée telle quelle au cas où le quart de jour de huit heures (D) serait séparé en deux quarts de quatre heures.

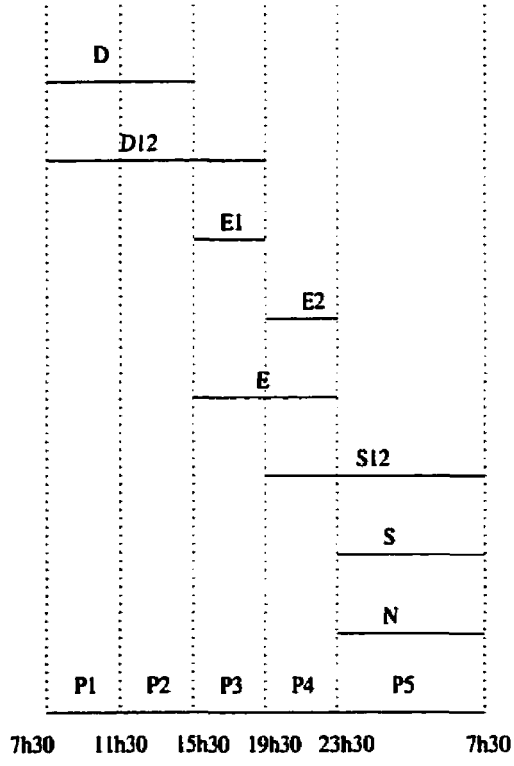


Figure 4.1: Périodes de quotas pour le centre des naissances

- L'horizon de planification couvre six semaines, soit quarante-deux jours de travail. Cet horizon est divisé en trois sous-périodes de deux semaines pour la contrainte qui contrôle la charge de travail affectée.

Type de journée	Période	Quota désiré	Déficit maximal	Surplus maximal
Lu-Ve	P1	9	1	1
Lu-Ve	P2	9	1	1
Lu-Ve	P3	8	1	1
Lu-Ve	P4	8	1	1
Lu-Ve	P5	7	1	1
Sa, Di et fériée	P1	7	1	1
Sa, Di et fériée	P2	7	1	1
Sa, Di et fériée	P3	7	1	1
Sa, Di et fériée	P4	7	1	1
Sa, Di et fériée	P5	7	1	1

Tableau 4.2: Contraintes de quotas (centre des naissances)

Les contraintes de quotas sont exprimées par périodes et sont de deux types distincts. Pour les jours de semaine (lundi à vendredi inclusivement), on doit avoir 9 infirmières en journée, 8 en soirée et 7 pendant la nuit. Les jours de fins de semaine et les jours fériés, on doit avoir 7 infirmières sur chaque période de quota. Le tableau 4.2 résume ces contraintes.

Globalement, cette unité présente un intérêt non négligeable : on dispose d'un nombre significatif d'infirmières pour faire tourner le programme, mais les différentes contraintes présentes ont pour effet de limiter la place mémoire occupée par l'instance et de rendre sa résolution tout à fait possible. Cette équipe d'infirmières présente les caractéristiques décrites ci-dessous.

- 50 infirmières sont présentes dans le fichier de test pour le centre des naissances.
- Une infirmière travaillant à plein temps doit effectuer une charge de travail de 80 heures aux deux semaines. Sur les 50 infirmières, 17 travaillent à plein

temps. Les autres effectuent des charges de travail comprises entre 30% et 80% de cette charge maximale de travail.

- Sur les 50 infirmières, 45 respectent la rotation de travail une fin de semaine sur deux. Deux autres ne travaillent jamais la fin de semaine. Les trois autres travaillent toutes les fins de semaine.
- Sur ces 50 infirmières, 16 doivent prendre des congés durant l'horizon de planification.

De plus, on dispose pour cette unité de soins de deux horizons consécutifs de planification, ce qui permet de tester le comportement du logiciel développé dans le cas de l'enchainement de deux horizons consécutifs. Les propriétés de l'unité énoncées plus haut sont constantes sur les deux horizons.

4.3.3 Données de l'unité "dialyses" (Hôpital Royal Victoria)

Dans cette sous-section nous présentons le dernier fichier de test reçu, qui concerne une unité de soins de l'hôpital Royal Victoria.

Cette unité a été choisie comme deuxième unité de test, car elle présente des caractéristiques très intéressantes pour observer le comportement du programme dans des cas plus particuliers. En effet, l'unité présente les caractéristiques décrites ci-dessous, certaines d'entre elles n'étant pas tout à fait dans le cadre standard imaginé lors de la conception du logiciel IRIS.

- L'horizon de planification consiste en six semaines, soit 42 jours. Les sous-périodes pour la contrainte de charge de travail ont une durée de 2 semaines.
- Cette unité est composée de 37 infirmières, ce qui représente une unité de taille moyenne.
- Les infirmières travaillent seulement six jours par semaine. En effet, personne ne travaille le dimanche dans cette unité. Les fins de semaines seront

donc réduites au samedi en ce qui concerne les rotations fins de semaine travaillées ou non.

- On dispose de cinq quarts de travail distincts ayant des durées de 8 ou 12 heures. Ces quarts de travail se recouvrent, il faut donc utiliser le concept de périodes de quotas pour décrire nos contraintes. Le schéma des périodes utilisées est le même que pour le centre des naissances (voir schéma 4.1).
- Les heures de travail vont de 7*h*30 à 23*h*30. La nuit n'est jamais travaillée au sein de cette unité. Là encore, il est intéressant de voir la versatilité de la modélisation qui permet de s'affranchir de telles contraintes. Les contraintes de quotas ne sont donc données que pour les périodes 1 à 4, la période 5 (nuit de 23*h*30 à 7*h*30) étant inutilisée.

Les contraintes de quotas sont résumées dans le tableau 4.3. Tous les jours, 13 infirmières doivent être présentes de 7*h*30 à 11*h*30, 16 doivent être là de 11*h*30 à 15*h*30, puis 12 de 15*h*30 à 19*h*30. Seule la période 4 diffère selon le jour de la semaine, puisqu'on a besoin de 8 infirmières de 19*h*30 à 23*h*30 en semaine (lundi à vendredi inclusivement) et de 5 la fin de semaine (samedi seulement puisque le dimanche n'est pas travaillé dans cette unité).

Les infirmières oeuvrant dans cette unité de soins présentent les caractéristiques décrites ici.

- La charge de travail normale est de 80 heures toutes les deux semaines. Sur les 37 infirmières de l'unité, 21 travaillent à temps plein. Les autres effectuent une charge de travail comprise entre 20% et 80% de la charge de travail nominale.
- Sur les 37 infirmières, 20 peuvent effectuer deux quarts de travail sur les cinq présents dans le fichier d'entrée, 16 en effectuent quatre et la dernière ne peut effectuer qu'un seul quart de travail.
- 33 des 37 infirmières suivent la rotation une fin de semaine travaillée sur

Journée	Période	Quota désiré	Déficit maximal	Surplus maximal
Semaine	P1	13	2	2
Semaine	P2	16	3	4
Semaine	P3	12	1	2
Semaine	P4	8	1	1
Samedi	P1	13	2	2
Samedi	P2	16	3	4
Samedi	P3	12	1	2
Samedi	P4	5	1	0

Tableau 4.3: Contraintes de quotas (unité "dialyses")

deux. Les 4 autres travaillent tous les samedis.

- 7 infirmières ont des congés à recevoir sur la période considérée.

4.4 Résultats et analyses

Dans cette section, nous étudions les résultats produits par le programme IRIS sur les jeux de données présentés dans la section précédente et les comparons à des résultats fournis par les hôpitaux pour les mêmes périodes.

Nous examinons dans un premier temps les résultats des tests effectués avec la collaboration du CHUM informatique, même si nous ne disposons d'aucun horaire de référence pour la période considérée.

Nous explorons plus en détails les tests effectués en collaboration avec le centre des naissances de l'hôpital Royal Victoria, qui présentent plus d'intêret puisqu'ils représentent un cas réel. De plus nous disposons des horaires produits par l'infirmière chef de ce centre pour les périodes de planification pour lesquelles les tests ont été réalisés.

Nous terminons avec le dernier jeu de test reçu, correspondant à l'unité "dia-

Nombre de solution réalisables trouvées	5
Optimum atteint	Oui
Temps d'exécution (minutes)	0.27
Valeur optimale	318.593

Tableau 4.4: Caractéristiques du déroulement du test (CHUM informatique)

lyses" de l'hôpital Royal Victoria. Nous ne disposons pas de l'horaire produit par l'infirmière chef pour ce test, mais pouvons néanmoins commenter la qualité des solutions obtenues.

Pour chaque horaire présenté, la valeur de la fonction objective correspondante est donnée. Précisons que ceci est à tire indicatif seulement, la valeur pouvant varier grandement d'une unité à l'autre suivant les différents coefficients entrés par les infirmières chefs. Le détail du calcul de cette valeur est expliqué par Vovor [2].

4.4.1 Données du CHUM informatique

Le fichier de test étant créé pour les expérimentations, on ne dispose d'aucun horaire généré de manière traditionnelle pour effectuer les comparaisons avec notre horaire. On ne pourra donc que regarder les résultats générés par le programme. Le fichier de résultats est présenté dans la figure 4.2.

Les caractéristiques de ce test sont résumées dans le tableau 4.4. On pourra noter que les tests se déroulent avec un nombre maximal de solutions réalisables conservées égale à 5. Le premier chiffre du tableau n'est donc pas réellement significatif.

La méthode étant exacte, toutes les contraintes sont vérifiées par toutes les solutions réalisables trouvées. Les résultats sont décrits dans le tableau 4.5. Cette table présente pour chaque quart le quota que l'on visait (identique pour tous les

Lot number / Numero de lot: 17
Task number / Numero de version: 1
Processing time / Temps de traitement: 0:0:27

Schedule number / Numero d'horaire: 1
Schedule value / Valeur de l'horaire: 318.593

Key for the shifts :

JOUR	8h00-16h00
SOIR	16h00-24h00
NUIT	24h00-8h00
VAC.	Vacancy granted for the day

Employee Number	Skill Level	Su Di 3-Oct	Mo Lu 4-Oct	Tu Ma 5-Oct	We Me 6-Oct	Th Je 7-Oct	Fr Ve 8-Oct	Sa Sa 9-Oct	Su Di 10-Oct	Mo Lu 11-Oct	Tu Ma 12-Oct	We Me 13-Oct	Th Je 14-Oct	Fr Ve 15-Oct	Sa Sa 16-Oct
1	1			NUIT	JOUR	JOUR	JOUR	JOUR	JOUR	SOIR	SOIR	SOIR	SOIR		
2	1			JOUR	JOUR	JOUR	JOUR	SOIR	SOIR	VAC.	JOUR	JOUR			
3	1	JOUR	JOUR		JOUR	JOUR		JOUR	JOUR	JOUR			JOUR	JOUR	
4	1		NUIT	JOUR			NUIT	NUIT	NUIT	NUIT	JOUR			NUIT	
5	1	JOUR	JOUR	SOIR			SOIR	SOIR	SOIR			NUIT	NUIT	SOIR	
6	1			NUIT	NUIT	NUIT	NUIT	JOUR	JOUR	JOUR	JOUR	JOUR	SOIR		
7	1	SOIR	SOIR	SOIR	SOIR	SOIR					NUIT	NUIT	JOUR	JOUR	

Employee Number	Skill Level	Su Di 17-Oct	Mo Lu 18-Oct	Tu Ma 19-Oct	We Me 20-Oct	Th Je 21-Oct	Fr Ve 22-Oct	Sa Sa 23-Oct	Su Di 24-Oct	Mo Lu 25-Oct	Tu Ma 26-Oct	We Me 27-Oct	Th Je 28-Oct	Fr Ve 29-Oct	Sa Sa 30-Oct
1	1			NUIT	JOUR	JOUR	JOUR	JOUR	JOUR	SOIR	SOIR	SOIR	SOIR		
2	1	JOUR	JOUR	JOUR	JOUR	SOIR					JOUR	JOUR	JOUR	JOUR	
3	1			JOUR		JOUR	JOUR	JOUR	JOUR	JOUR	JOUR	JOUR	JOUR		
4	1		NUIT	JOUR			NUIT	NUIT	NUIT	NUIT			NUIT	SOIR	
5	1	SOIR	SOIR			NUIT	SOIR	SOIR	SOIR			NUIT	JOUR	JOUR	
6	1			NUIT	NUIT	NUIT	NUIT	JOUR	JOUR	JOUR	SOIR	SOIR	SOIR		
7	1		JOUR	SOIR	SOIR	SOIR					NUIT	NUIT	NUIT	NUIT	

Figure 4.2: Résultats obtenus pour le CHUM informatique

Quart	Quota visé	Déficit max. demandé	Déficit max. affecté	Surplus max. demandé	Surplus max. affecté
JOUR	3	1	1	1	0
SOIR	2	1	1	1	0
NUIT	2	1	1	1	0

Tableau 4.5: Caractéristiques de la solution (CHUM informatique)

jours de l'horizon dans ce cas particulier), le déficit maximal accordé ainsi que le surplus maximal.

Disposant d'une équipe de sept infirmières et ayant des requêtes de quotas pour sept infirmières par jour, la solution optimale ne pouvait contenir aucun surplus d'infirmière. C'est ce qui est vérifé ici. D'autre part, on peut constater que les requêtes de quotas sont dans ce test relativement contraignantes puisque le programme n'a pu affecter la valeur cible que sur 25% des jours de l'horizon pour le quart de jour, et sur 21.43% sur les quarts de nuit.

4.4.2 Données du centre des naissances

Les résultats obtenus sur les fichiers de test du centre des naissances de l'hôpital Royal Victoria sont très encourageants. Nous présentons dans un premier temps l'horaire produit par l'infirmière chef du centre des naissances pour la première période de planification, puis nous examinons ensuite l'horaire produit par le programme IRIS pour la même période. Nous montrons enfin comment le programme s'est comporté dans le cas de l'enchaînement de deux horizons consécutifs.

4.4.2.1 Analyse de l'horaire produit par Royal Victoria

A des fins de comparaison et d'évalution du programme, nous avons demandé à l'hôpital Royal Victoria de nous fournir l'horaire correspondant à la période de

Journée	Période	Quota visé	Déficit max. demandé	Déficit max. affecté	Surplus max. demandé	Surplus max. affecté
Lu-Ve	P1	9	1	1	1	2
Lu-Ve	P2	9	1	1	1	2
Lu-Ve	P3	8	1	1	1	3
Lu-Ve	P4	8	1	0	1	2
Lu-Ve	P5	7	1	1	1	2
Sa, Di et fériée	P1	7	1	1	1	3
Sa, Di et fériée	P2	7	1	1	1	3
Sa, Di et fériée	P3	7	1	1	1	5
Sa, Di et fériée	P4	7	1	1	1	4
Sa, Di et fériée	P1	7	1	2	1	1

Tableau 4.6: Caractéristiques de l'horaire fourni (centre des naissances)

planification retenue pour les tests. Nous avons analysé cet horaire et avons ainsi constaté les qualités et défauts de celui-ci.

L'horaire produit par l'infirmière chef responsable du centre des naissances de l'hôpital Royal Victoria respecte toutes les contraintes intrinsèques relatives à chaque infirmière. Cependant, il ne respecte pas les contraintes de quotas que l'on nous avait spécifiées pour les tests (voir tableau 4.2). On avait plutôt les résultats présentés dans le tableau 4.6.

Pour cette unité, les contraintes de quotas ont été initialement données avec une tolérance de plus ou moins une infirmière sur chaque contrainte. Un rapide coup d'oeil au tableau 4.6 nous montre que ces contraintes ne sont pas respectées par l'horaire produit par l'infirmière chef responsable de cette unité. Dans le but de valider l'outil logiciel développé, nous avons effectué deux séries de tests. Le premier test a été effectué avec les contraintes de quotas souhaitées par l'infirmière

Nombre de solution réalisables trouvées	0
Optimum atteint	Non
Temps d'exécution (minutes)	environ 240

Tableau 4.7: Caractéristiques du test (centre des naissances - quotas initiaux)

Nombre de solution réalisables trouvées	1
Optimum atteint	Oui
Temps d'exécution (minutes)	318
Valeur optimale	1936.18

Tableau 4.8: Caractéristiques de la solution optimale (centre des naissances - période 1)

chef, mais IRIS n'a pas trouvé de solution réalisable à ce problème (voir table de résultats 4.7). Nous avons donc modifié les contraintes de quotas pour résoudre le même problème que les hôpitaux, et IRIS a généré une solution meilleure que celle trouvée par l'infirmière chef en termes de satisfaction des contraintes de quotas.

4.4.2.2 Analyse de l'horaire produit par IRIS

Première période

Le fichier résultat produit par IRIS pour la première période est présenté dans les figures 4.3 et 4.4. Ce résultat correspond à la solution optimale du problème. Cette solution et la preuve d'optimalité asociée ont nécessité un peu plus de cinq heures de temps de calcul (voir table 4.8).

La solution trouvée est jugée très satisfaisante par Marie-France Noëlle, infirmière responsable de l'unité "centre des naissances" de l'hôpital Royal Victoria. Cette solution ne présente en effet aucun défaut majeur, c'est-à-dire qu'elle respecte toutes les contraintes spécifiées par les hôpitaux lors de la définition initiale

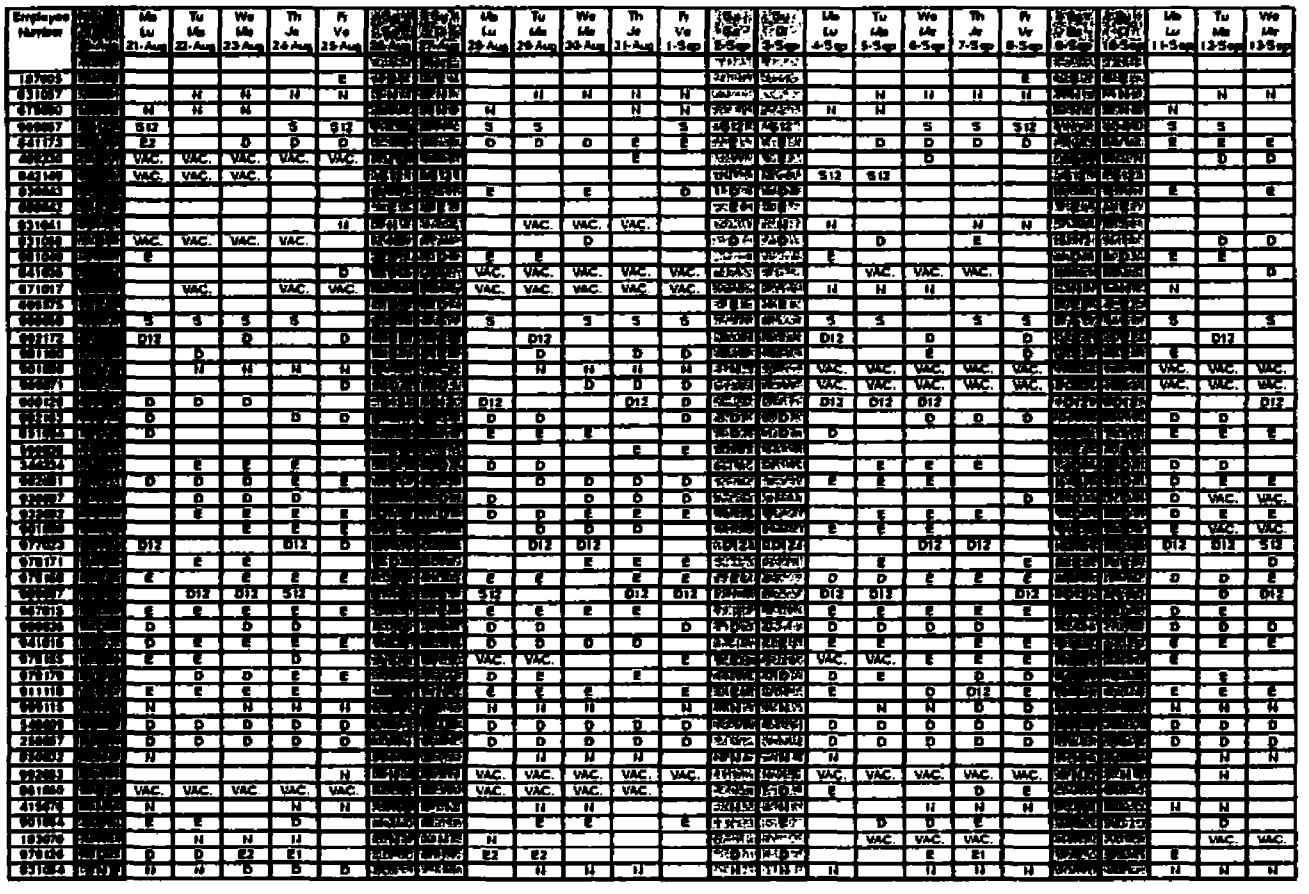


Figure 4.3: Résultats obtenus pour le centre des naissances (période 1)

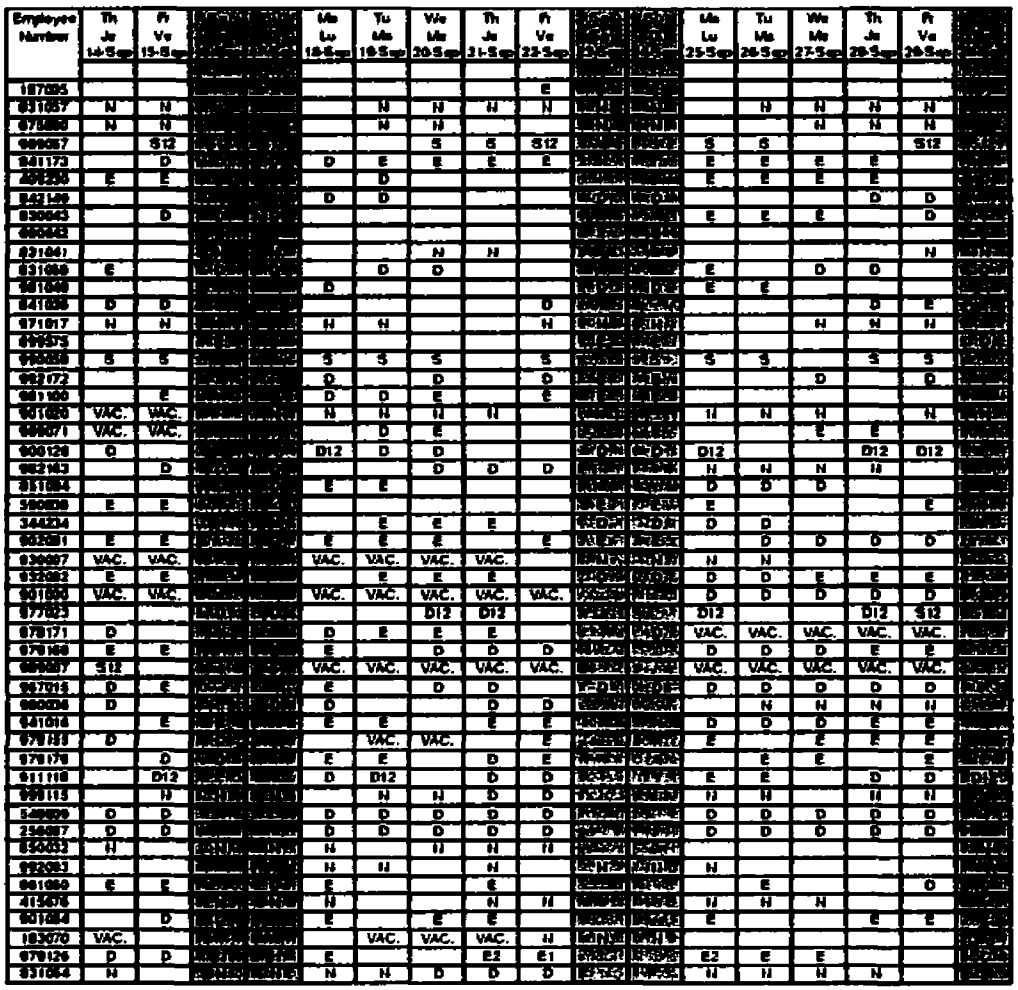


Figure 4.4: Résultats obtenus pour le centre des naissances (période 1 - suite)

Journée	Période	Quota visé	Déficit max. demandé	Déficit max. affecté	Surplus max. demandé	Surplus max. affecté
Lu-Ve	P1	9	1	1	2	2
Lu-Ve	P2	9	1	1	2	2
Lu-Ve	P3	8	1	1	3	3
Lu-Ve	P4	8	1	1	2	2
Lu-Ve	P5	7	1	1	3	3
Sa, Di ou fériée	P1	7	1	1	3	2
Sa, Di ou fériée	P2	7	1	1	3	2
Sa, Di ou fériée	P3	7	1	1	5	2
Sa, Di ou fériée	P4	7	1	1	4	3
Sa, Di ou fériée	P1	7	2	1	1	0

Tableau 4.9: Caractéristiques de l'horaire produit (centre des naissances - période 1)

du problème. De plus, le programme a souvent fait mieux que l'infirmière chef dans le respect des contraintes de quotas, puisque bien souvent on dépasse les quotas demandés (voir table 4.9).

Comme on peut le constater, la plupart du temps, le programme IRIS a eu tendance à affecter plus d'infirmières que l'infirmière chef. Ceci s'explique par le fait que certaines infirmières (trois pour être exact) devraient être en absence pour de la formation sur une partie de l'horizon. Mais ces infirmières doivent être payées pendant les périodes de formation. Le programme ne gérant pas les absences pour formation, les trois infirmières sont marquées comme travaillant pendant tout l'horizon dans le fichier d'entrée. Le programme affecte donc à ces infirmières des horaires, alors que l'infirmière chef ne les avait pas prises en compte.

Lors de l'examen de l'horaire généré par IRIS par l'infirmière chef, quelques remarques sur la qualité de l'horaire généré ont été formulées et sont listées ci-bas. Précisons que tous ces problèmes proviennent de contraintes qui avaient été mal définies ou mal cernées lors de la phase de définition de la problématique à traiter. Pour chaque imperfection, une solution est donnée.

- *Problème 1 :* Comme vu plus haut, sauf exceptions dues aux vacances, une fin de semaine ne doit jamais être brisée.

  Solution : Pour résoudre ce problème, les règles de construction des arcs doivent être revues en présence de jours de congé. En particulier, si une semaine est candidate pour recevoir des jours de congé et si la fin de semaine auparavant est travaillée, seuls des arcs partant d'un quart de travail le samedi et arrivant à un noeud de congé pendant la semaine sont autorisés. Le même raisonnement est valide pour les fins de semaine travaillées après des congés (rappelons que dans ce cas, seul le dimanche est travaillé). Cependant, en raison de variantes pouvant intervenir dans certaines unités (dimanche systématiquement non travaillé entre autres) il faudra penser à une modélisation faisant intervenir des noeuds fictifs pour résoudre tous les cas de figure possibles.

- *Problème 2 :* Lorsque l'infirmière travaille à temps plein, c'est-à-dire 80 heures sur une période de deux semaines, elle préfère en général effectuer six quarts de travail de douze heures et un quart de travail de huit heures.

  Solution : Cette préférence, que l'on pourrait faire intervenir comme une contrainte molle, n'avait jamais été mentionnée avant les premiers tests effectués à la fin de l'été. Il s'agit donc d'un point à ajouter lors d'une version subséquente du programme. Une solution est de pénaliser les chemins qui ne respectent pas un schéma donné de répartition des quarts de travail.

- *Problème 3 :* L'infirmière chef se plaignait que certaines infirmières ne res-

pectaient pas la rotation des fins de semaine (lorsqu'une infirmière travaille une fin de semaine sur deux, parfois la mauvaise fin de semaine est affectée). Cependant, ceci s'explique par le fait que l'infirmière chef ne nous donne aucune information sur l'horaire pour l'horizon précédent la période de planification courante. Ainsi, le programme ne peut savoir où l'infirmière en est rendue dans ses rotations de fins de semaine.

Solution : L'infirmière chef doit nous fournir les données concernant les affectations reçues pendant l'horizon précédent pour que les enchaînements des deux horizons soient corrects. Comme nous le verrons un peu plus loin, pour la deuxième période fournie, cette remarque n'a plus lieu d'être puisque l'on dispose de l'horizon précédent.

Par ailleurs, sur cet exemple particulier, on peut observer (table 4.8) que la solution optimale correspond à la première solution réalisable trouvée par le programme. Cette solution a été obtenue en moins de deux minutes de temps de calcul, et a été sortie grâce à une procédure d'arrondi en nombres entiers mise au point lors du développement du logiciel (voir section 3.2).

Cette procédure permet pour les grandes instances de sauver énormément de temps. En effet, sans cette procédure, le programme n'arrive pas à trouver de solution réalisable pour le jeu de données considéré en deçà de deux heures de temps de calcul, mais avec cette procédure il trouve une solution réalisable en moins de deux minutes. D'où un gain très appréciable pour les infirmières, puisque la rapidité de calcul est le critère qui retient l'attention des infirmières chefs.

Précisons que les infirmières des unités partenaires souhaitent établir une limite de temps de deux heures pour l'exécution. Comme on peut le constater, cette limite aurait tout de même permis de trouver la solution optimale, mais aurait interrompu la preuve de l'optimalité de la solution qui a été effectuée par le programme en un peu plus de cinq heures.

Nombre de solution réalisables trouvées	2
Optimum atteint	Oui
Temps d'exécution (minutes)	269
Valeur optimale	1769.44

Tableau 4.10: Caractéristiques de la solution optimale (centre des naissances - période 2)

D'un point de vue plus mathématique, ce premier test et les résultats qui l'accompagnent sont très encourageants, car l'horaire trouvé en un temps très raisonnable respecte bien toutes les contraintes que nous avions implantées dans le programme.

Deuxième période

Pour la deuxième période de test fournie, qui est calendairement consécutive à la première, les résultats de calcul sont résumés dans la table 4.10. Le meilleur horaire produit pour cette période (la solution optimale du problème) est donné dans les figures 4.5 et 4.6. Les contraintes de quotas pour cette période étaient légèrement différentes de celles de la période précédente (voir table 4.11).

Le test avec cette deuxième période s'avère très concluant, puisqu'il montre que le programme se comporte très bien lors de la succession de deux horizons : les contraintes de fins de semaine et de rotation de type de quarts sont parfaitement respectées d'un horizon à l'autre. Les seules difficultés rencontrées sont lorsqu'une contrainte intrinsèque à une infirmière donnée change de formulation (en termes de bornes de réalisabilité) lors du passage d'un horizon à l'autre. Par exemple, on peut penser à une infirmière qui travaille une fin de semaine sur deux lors de la première rotation, mais toutes les fins de semaines lors de la seconde. La solution apportée dans ce cas est de ne considérer pour le passage du premier au deuxième horizon que les valeurs les moins contraignantes, de façon à assurer une

Lot number / Numero de lot: 6230
Task number / Numero de vacation: 1
Processing time / Temps de traitement: 4.269.16143

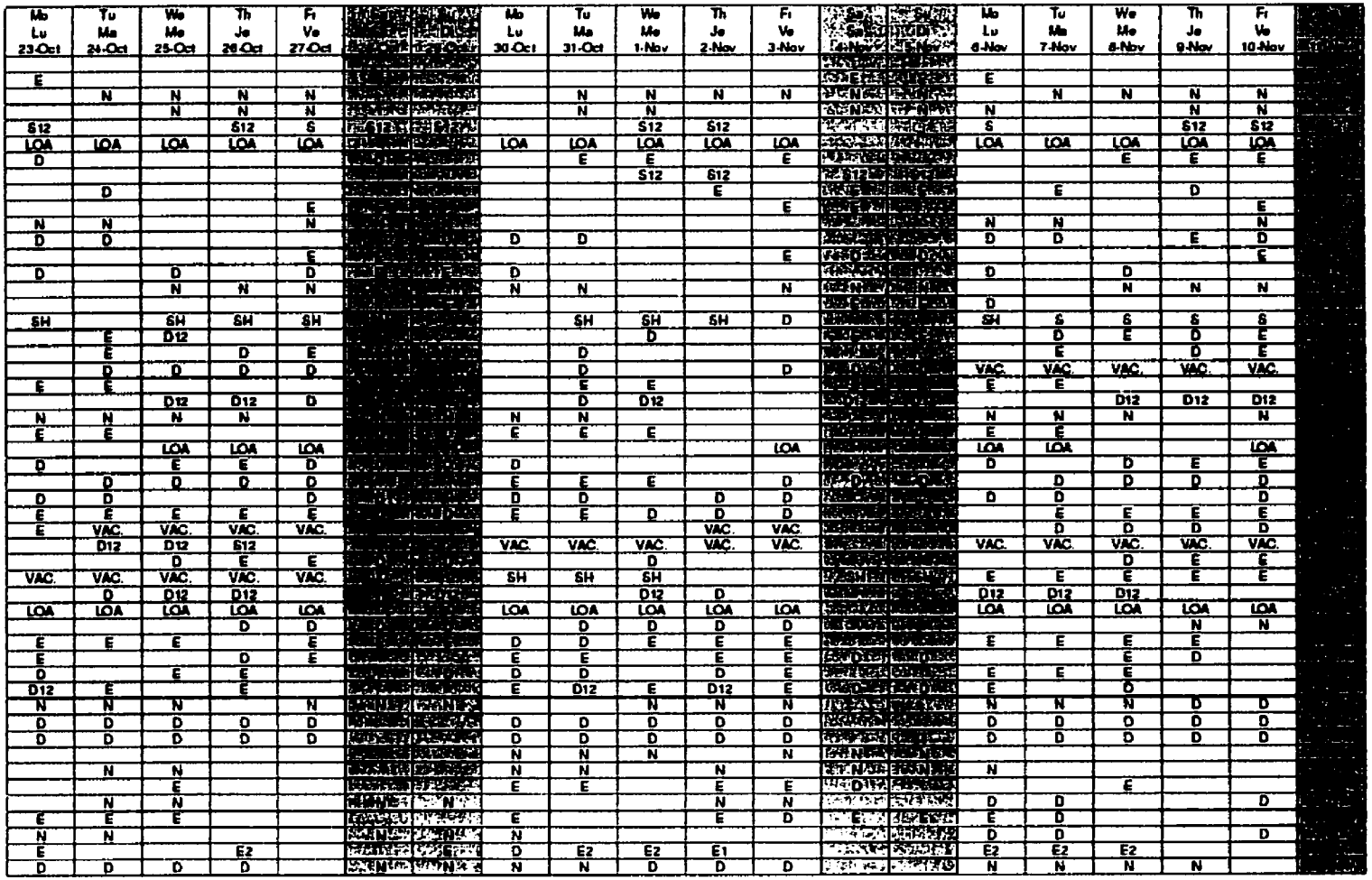
Schedule number / Numero d'horaire: 1
Schedule value / Valeur de l'horaire: 1788.44

Key for the shifts:

D	7h30-15h30	E	15h30-23h30	VAC.	Vacations granted
D12	7h00-19h30	S	23h30-7h30	Others	Non worked days (various reasons)
E1	15h30-18h00	N	23h30-7h30		
E2	19h30-23h30	S12	19h30-7h00		

Employee Number		Mo Lu 3-Oct	Tu Ma 3-Oct	We Me 4-Oct	Th Je 5-Oct	Fr Ve 6-Oct			Mo Lu 9-Oct	Tu Ma 10-Oct	We Me 11-Oct	Th Je 12-Oct	Fr Ve 13-Oct			Mo Lu 16-Oct	Tu Ma 17-Oct	We Me 18-Oct	Th Je 19-Oct	Fr Ve 20-Oct	
187005									E												
831057			N	N	N	N				N	N	N	N				N	N	N	N		
675860			N	N							N	N	N				N	N				
886067				S12	S12				S12			S12	S						S12	S12		
861173		LOA	LOA	LOA	LOA	LOA			LOA	LOA	LOA	LOA	LOA			LOA	LOA	LOA	LOA	LOA		
409230		E		E		E			E	D		D	E			D	D	D	D			
842149				S12	S12													S12	S12			
800043			E		E					D							E		E			
880842						E							E							E		
831041									N	N			N									
831068		D	D						D	D		E	E			VAC	VAC	VAC	VAC	VAC		
881048						E							E							E		
841038									D		D		D									
971017		N	N			N					N	N	N			N	N			N		
886878						D																
880088			DP	DP	DP	DP			SH	DP		SH	SH			DP	DP	DP	DP	DP		
882172				D12		D				E	D12							D12	E	E		
881100			D							E		E	E				D					
901030		D		D	D	D				D	D	D	D			D		D	D	D		
886071			E	E	E				E	E							E	E				
800188		D	D12			D12					D12	D12	D			D	D12			D12		
882182		N	N						N	N	N	N				N	N					
891094		E	E						E	E	E					E	E					
880838						LOA					LOA	LOA	LOA							LOA		
344234		D							D		E	E	D			D						
902081		E	E	E					E	D	D	D	D			E	E	E		D		
830007		D			D	D			D	D			D			D	D		D	D		
932082			E	E					D	VAC	VAC	VAC	VAC	VAC		D		D	D	D		
901000									D	D	E	E	E	E								
877023			D12	D12	S12				D12	D12			D	D12			D12	D12			D12		
878171				VAC	VAC	VAC								E					D			
879169		VAC	VAC	VAC	VAC	VAC			VAC	VAC	VAC	VAC	VAC			VAC	VAC	VAC	VAC	VAC		
889007				D12	D12				D12	D12	D12					D12		D12	D12	D12		
887015		LOA	LOA	LOA	LOA	LOA			LOA	LOA	LOA	LOA	LOA			LOA	LOA	LOA	LOA	LOA		
880038		D	D						D	D	D	D				D	D	D	D	D		
841016		E	D	D	D	D				E	E	D	D			E	E	E	E	E		
879155				D		D			VAC	VAC	VAC	VAC	VAC			E		E	E			
879179		E		D	E	E			E		D	E				E	E	E	D	E		
911118		VAC	VAC	VAC	VAC	VAC					E	D12				E		D12	E	E		
888115			N	N	D	D			D	D		N	N			N	N	N	D	D		
548609		D	D	D	D	D			D	D	D	D	D			D	D	D	D	D		
256087		D	D	D	D	D			D	D	D	D	D			D	D	D	D	D		
850032		N	N	N		N										N	N	N		N		
882083		N				N				N	N					N				N		
881080		E	E		E	E				E						E	E			E		
415676		N			N	N			N	N						N			N	N		
901064		E	E	E	D	E					E						E			E		
183070		N							N	N	N					N		N		N		
879128		E1	D	E2	E	E2				E2	E2	E2				D	D	E2	E			
831094		N	N	D	D	D				N	N	N				N		N	N	N		

Figure 4.5: Résultats obtenus pour le centre des naissances (période 2)



Mo Lu 23-Oct	Tu Ma 24-Oct	We Me 25-Oct	Th Je 26-Oct	Fr Ve 27-Oct	[illegible]	[illegible]	Mo Lu 30-Oct	Tu Ma 31-Oct	We Me 1-Nov	Th Je 2-Nov	Fr Ve 3-Nov	[illegible]	[illegible]	Mo Lu 6-Nov	Tu Ma 7-Nov	We Me 8-Nov	Th Je 9-Nov	Fr Ve 10-Nov
E														E				
	N	N	N	N				N	N	N	N				N	N	N	N
		N	N	N				N	N					N			N	N
S12			S12	S					S12	S12				S			S12	S12
LOA	LOA	LOA	LOA	LOA			LOA	LOA	LOA	LOA	LOA			LOA	LOA	LOA	LOA	LOA
D								E	E		E					E	E	E
									S12	S12								
	D									E					E		D	
				E							E							E
N	N			N										N	N			N
D	D						D	D						D	D		E	D
				E							E							E
D		D		D			D							D		D		
		N	N	N			N	N			N					N	N	N
														D				
SH		SH	SH	SH				SH	SH	SH	D			SH	S	S	S	S
	E	D12							D						D	E	D	E
	E		D	E				D							E		D	E
	D	D	D	D				D			D			VAC.	VAC.	VAC.	VAC.	VAC.
E	E							E	E					E	E			
		D12	D12	D				D	D12							D12	D12	D12
N	N	N	N				N	N						N	N	N		N
E	E						E	E	E					E	E			
		LOA	LOA	LOA							LOA			LOA	LOA			LOA
D		E	E	D			D							D		D	E	E
	D	D	D	D			E	E	E		D				D	D	D	D
D	D			D			D	D		D	D			D	D			D
E	E	E	E	E			E	E	D	D	D				E	E	E	E
E	VAC.	VAC.	VAC.	VAC.						VAC.	VAC.				D	D	D	D
	D12	D12	S12				VAC.	VAC.	VAC.	VAC.	VAC.			VAC.	VAC.	VAC.	VAC.	VAC.
		D	E	E					D							D	E	E
VAC.	VAC.	VAC.	VAC.	VAC.			SH	SH	SH					E	E	E	E	E
	D	D12	D12						D12	D				D12	D12	D12		
LOA	LOA	LOA	LOA	LOA			LOA	LOA	LOA	LOA	LOA			LOA	LOA	LOA	LOA	LOA
			D	D				D	D	D	D						N	N
E	E	E		E			D	D	E	E	E			E	E	E	E	
E			D	E			E	E		E	E					E	D	
D		E	E				D	D		D	E			E	E	E		
D12	E		E				E	D12	E	D12	E			E		D		
N	N	N		N					N	N	N			N	N	N	D	D
D	D	D	D	D			D	D	D	D	D			D	D	D	D	D
D	D	D	D	D			D	D	D	D	D			D	D	D	D	D
							N	N	N		N							
	N	N					N	N		N				N				
		E					E	E		E	E					E		
	N	N								N	N			D	D			D
E	E	E					E			E	D			E	D			
N	N						N							D	D			D
E			E2				D	E2	E2	E1				E2	E2	E2		
D	D	D	D				N	N	D	D	D			N	N	N	N	

Figure 4.6: Résultats obtenus pour le centre des naissances (période 2 - suite)

Journée	Période	Quota visé	Déficit max. demandé	Déficit max. affecté	Surplus max. demandé	Surplus max. affecté
Lu-Ve	P1	9	2	0	5	4
Lu-Ve	P2	9	2	0	5	4
Lu-Ve	P3	8	2	1	5	2
Lu-Ve	P4	8	2	1	5	1
Lu-Ve	P5	7	2	1	5	1
Sa, Di ou fériée	P1	7	1	1	5	1
Sa, Di ou fériée	P2	7	1	1	5	1
Sa, Di ou fériée	P3	7	1	1	5	1
Sa, Di ou fériée	P4	7	1	1	5	1
Sa, Di ou fériée	P1	7	1	1	5	1

Tableau 4.11: Caractéristiques de l'horaire optimal produit (centre des naissances - période 2)

continuité dans les horaires, puis de reprendre les nouvelles valeurs pour le reste de la deuxième période.

Les tests pour cette période fournissent un horaire réalisable et un horaire optimal. En termes de fonction objective, la différence de qualité est de 9%. Par contre, la différence est plus sensible sur certaines contraintes, notamment au niveau du respect des contraintes de quotas. La solution optimale est constitué d'horaires qui, pris ensemble, forment des quotas d'infirmières plus proches des valeurs cibles. En effet, la solution optimale du problème est composée d'emplois du temps qui respectent exactement la valeur cible des contraintes de quotas dans 30% des cas, tandis que la première solution réalisable trouvée ne les respecte exactement que dans 20% des cas.

4.4.3 Données de l'unité de dialyses

Nous allons dans cette section présenter la série de résultats obtenus pour le troisième jeu de tests reçu, celui de l'unité de dialyses. Nous ne disposons pas de l'horaire original produit par l'infirmière chef pour cette période mais il est néanmoins intéressant de regarder les caractéristiques des résultats obtenus.

Précisons que malgré le nombre moins important d'infirmières présentes dans cette unité par rapport au centre des naissances, l'espace mémoire occupé par le programme IRIS était plus important et ne nous a pas permis d'obtenir la solution optimale pour ce problème. Ceci s'explique par le fait que pour le centre des naissances, la plupart des infirmières n'ont que deux quarts de travail réalisables. Pour cette unité, plus de la moitié des infirmières ont quatre quarts réalisables, ce qui fait augmenter la taille des graphes. Nous avons donc dû nous contenter de solutions réalisables.

Plusieurs tests ont été réalisés. Le premier est assez décevant et les résultats sont présentés dans la table 4.12.

Comme le lecteur peut le constater, le premier test effectué n'a pas trouvé de

Nombre de solution réalisables trouvées	0
Optimum atteint	Non
Temps d'exécution (minutes)	environ 75

Tableau 4.12: Caractéristiques du déroulement du test 1 (unité "dialyses")

Journée	Période	Quota visé	Déficit max. demandé	Déficit max. affecté	Surplus max. demandé	Surplus max. affecté
Lu-Ve	P1	13	4	2	5	2
Lu-Ve	P2	16	5	3	7	1
Lu-Ve	P3	12	3	2	5	4
Lu-Ve	P4	8	3	3	4	3
Sa	P1	13	4	2	5	1
Sa	P2	16	5	3	6	0
Sa	P3	12	3	2	5	0
Sa	P4	5	3	0	3	3

Tableau 4.13: Caractéristiques de l'horaire généré (unité "dialyses")

solution réalisable au problème et a été abandonné (dû à des problèmes d'occupation mémoire) au bout de $1h15$ de temps de calcul. Il a donc fallu, pour tenter d'obtenir un horaire pour cette unité, relaxer les contraintes de quotas. Ceci a été effectué de sorte que les contraintes de quotas étaient celles présentées dans le tableau 4.13.

Grâce aux intervalles de réalisabilité élargis pour les périodes de quota, le programme IRIS a très facilement trouvé une solution réalisable (voir table 4.14) qui est de très bonne qualité au regard du respect des contraintes de quotas (voir table 4.13).

Comme on peut le constater, la solution trouvée respecte presque les con-

Nombre de solution réalisables trouvées	1
Optimum atteint	Non
Temps d'exécution (minutes)	2.14

Tableau 4.14: Caractéristiques du déroulement du test 2 (unité "dialyses")

traintes de quotas originalement données par les hôpitaux. Seules les contraintes sur les fins de semaine empêchaient de trouver une solution réalisable lors du premier jeu de test. Ceci peut s'expliquer par la grande proportion d'infirmières effectuant une rotation stricte une fin de semaine travaillée sur deux. En effet, le programme a bien respecté cette contrainte, mais le fichier d'entrée était fourni sans l'historique précédent. Il a donc fallu modifier l'historique des infirmières dans le fichier d'entrée pour leur permettre de respecter la rotation des fins de semaine. Il est donc très probable que cette modification ait introduit un biais dans le jeu de tests empêchant l'outil IRIS de trouver une solution réalisable avec la première version du fichier d'entrée.

Globalement, ces résultats sont satisfaisants puisque, là encore, ils démontrent que l'infirmière chef peut obtenir un résultat tout à fait satisfaisant en quelques minutes pour une unité de 37 infirmières. De plus, l'intégration de plus en plus raffinée de l'outil IRIS va permettre (notamment grâce à la prise en compte des historiques) de traiter de mieux en mieux la plupart des problèmes proposés.

Conclusion

La génération d'horaires est un sujet traité par de nombreux auteurs, tant sur le plan théorique qu'en collaboration pratique avec des industriels. Plusieurs compagnies ont même été fondées à partir de cette volonté d'automatisation de la génération de ces horaires. Citons les exemples d'AD OPT et GIRO, qui sont deux compagnies montréalaises issues du milieu universitaire. AD OPT [21] fait affaire en particulier dans le domaine de la planification d'horaires de personnels dans le transport aérien, tandis que GIRO [22] développe des solutions axées autour du transport urbain.

Comme on a pu le constater, plusieurs méthodes et plusieurs modélisations existent pour tenter de solutionner le problème dans le cas particulier de la génération d'horaires pour des infirmières. Certains chercheurs exploitent le fait que dans certains hôpitaux les horaires soient cycliques, d'autres génèrent des horaires n'ayant aucune propriété temporelle apparente. Les méthodes utilisées sont également diverses. Comme on l'a vu, nous avons choisi une méthode basée sur la programmation linéaire et qui fait appel à plusieurs algorithmes classiques en recherche opérationnelle. Cette méthode présente l'avantage de permettre une certaine généricité, puisque le même outil pourra être utilisé dans plusieurs hôpitaux.

Cependant, cette généricité se traduit par une modélisation complexe, notamment au niveau du problème auxiliaire. Celui-ci se formule comme un problème de plus court chemin dans un graphe acyclique avec contraintes de ressources sur les arcs. Ce sont ces ressources qu'il faut prendre soin de définir proprement. En ef-

fet, une modélisation incorrecte des différents ressources aurait pour conséquence de pouvoir obtenir soit des horaires non réalisables, soit l'élimination d'horaires réalisables par le programme. Il est donc important de passer au travers de la modélisation des ressources pour se convaincre qu'aucun cas n'a été oublié lors de la définition de cette partie du problème.

C'est d'ailleurs pour cela que certaines contraintes ne sont pas prises en compte dans la version finale de l'outil développé. Comme on a pu le constater, au fur et à mesure des tests, on s'est aperçu que les hôpitaux partenaires avaient oublié de mentionner certaines contraintes qu'ils aimeraient voir prises en compte. La modélisation de ces contraintes a été effectuée, mais pas leur implémentation, qui sera effectuée dans une version subséquente du logiciel. De la même manière, un souci de simplification nous a amené à redéfinir la modélisation de la ressource gérant les jours de congés.

Enfin, on a pu constater que les résultats de calcul sont très satisfaisants, car ils permettent de valider l'outil développé. Soulignons encore une fois que les hôpitaux partenaires sont plus intéressés pour le moment à obtenir des solutions réalisables en temps très court qu'à obtenir la solution optimale et la preuve d'optimalité associée. Ceci leur permet en effet de tester différentes configurations d'horaires et de choisir celui qui leur paraît être le meilleur.

Il est important de garder en ligne de mire que l'objectif du logiciel développé est d'être opérationnel et utilisable par les infirmières. Une première version avait d'ailleurs été développée pour montrer la faisabilité de ce type d'outils. La version courante d'IRIS dont le développement est décrit dans le présent mémoire est à peu de choses près celle qui va être installée dans les différentes unités de soin. Il est donc important que tout en respectant le cahier des charges défini par l'Ecole et les hôpitaux, elle respecte également les souhaits des principaux utilisateurs. C'est d'ailleurs cette version qui va être implantée dans la plupart des unités de soins du MUCH, le regroupement d'hôpitaux dépendants de l'université McGill.

Bien entendu, le développement de l'outil est en constante évolution, et de nombreux ajouts viendront se greffer pour le rendre de plus en plus performant. On peut évidemment penser à l'addition d'une interface graphique, souhait exprimé par les infirmières pour rendre l'outil plus convivial et d'approche plus facile. Mais il y a également, à terme, l'abandon de la librairie ABACUS, qui gère le schéma de résolution de notre problème, mais est très gourmande en ressources (mémoire et temps de calcul). Le travail sur ce type de projet est donc loin d'être terminé, et même si ce mémoire a permis de valider le coeur du logiciel en développement, de nombreux défis restent à relever.

Bibliographie

[1] B. Jaumard; F. Semet et T. Vovor. Generalized Linear Programming Model for Nurse Scheduling. *Europen Journal Of Operational Research*, 107(1):1–18, May 1998.

[2] T. Vovor. *Problèmes de chemins bicritères ou avec contraintes de ressource : algorithmes et applications.* Thèse de doctorat, Ecole Polytechnique de Montréal, 1997.

[3] R. Hung. Hospital Nurse Scheduling. *Journal of Nursing Administration*, 25:21–23, 1995.

[4] L.D. Smith et A.Wiggins. A Computer Based Scheduling System. *Computers and Operations Research*, 4:195–212, 1977.

[5] H.E. Miller; W.P. Pierskalla et G.J. Rath. Nurse Scheduling Using Mathematical Programming. *Operations Research*, 24:857–870, 1976.

[6] D.M. Warner. Scheduling Nursing Personnel According to Nursing Preference: a Mathematical Programming Approach. *Operations Research*, 24:842–856, 1976.

[7] S. Abdennadher et H. Schlenker. Nurse Scheduling using Constraint Programming. In *Proceedings of the 1999 National Conference On Artificial Intelligence (AAAI-99)*, pages 838–843, July 1999.

[8] G. Weil; K. Heus; P. François et M. Poujade. Constraint Programming for Nurse Scheduling. *IEEE Engineering in Medicine and Biology*, 14(4):417–

422, Jul-Aug 1995.

[9] K.A. Dowsland. Nurse Scheduling with Tabu Search and Strategic Oscillation. *European Journal Of Operational Research*, 106(2-3):393–407, April 1998.

[10] G.L. Nemhauser et L.A. Wolsey. *Integer and Combinatorial Optimization.* Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley-Interscience, 1988.

[11] L. Wolsey. *Integer Programming.* Wiley, 1998.

[12] G. Desaulniers; J. Desrosiers; I. Ioachim; M.M. Solomon; F. Soumis et D. Villeneuve. A Unified Framework for Deterministic Time Constrained Vehicle Routing and Crew Scheduling Problems. Technical report G-94-46, GERAD, Canada, H3T 1V6, 1994.

[13] J.C. Foster D.M. Ryan. An Interger Programming Approach to Scheduling. *Computer Scheduling of Public Transport*, pages 269–280, 1981.

[14] V. Chvátal. *Linear Programming.* Freeman, 1983.

[15] M. Desrochers. An Algorithm for the Shortest Path Problem with Resource Constraints. Technical report G-88-27, GERAD, Canada, H3T 1V6, 1988.

[16] P. Galinier; B. Jaumard et P. Labit. Horaires d'infirmières : description des formats d'entrées/sorties. Août 2000.

[17] Hôpital Royal Victoria. Convention collective du personnel infirmier.

[18] M. Jünger et G. Reinelt. ABACUS, a Branch and Cut System, Version 2.2, User's Guide and Reference Manual, octobre 1998.

[19] OREAS. Site web http://www.oreas.de.

[20] CPLEX division d'ILOG. Site web http://www.cplex.com.

[21] AD OPT. Site web http://www.ad-opt.com.

[22] GIRO. Site web http://www.giro.ca.

Annexe A

Exemple de fichier d'entrée : extrait du fichier d'entrée du CHUM informatique

Dans cette annexe nous présentons un extrait du fichier d'entrée fourni par le CHUM informatique à des fins de test. Les contraintes de quotas ont été coupées (seule deux contraintes sont présentées pour chaque type de quart, alors que dans le fichier réel il y en a une pour chaque jour et chaque quart), et une seule infirmière est présente (il y en a 7 dans le fichier réel).

```
Info_generales
Numero_de_lot..........................  17
Numero_de_version......................  1
Nombre_d'infirmieres...................  7
Date_de_debut..........................  1999/10/03
Date_de_fin............................  1999/10/30
Nombre_de_titres_d'emploi..............  1
Nombre_de_periodes_de_quotas_par_jour...  1
Nombre_de_quarts_possibles.............  3

Preferences(+)_et_aversions(-)
Poids_de_la_composante_salaire(%_entre_0_et_100).... 0
Poids_de_la_composante_Preference(%entre_0_et_100).. 100
```

```
Aversion(-)
pour_surplus_de_quotas.............................. -100
pour_deficits_de_quotas............................. -100
non_respect_des_fins_de_semaine..................... -100

Autres_directives_de_groupe
Titre_d'emploi    Quarts_de_travail   Periode_de_quota Jours    Preference_ou_aversion
nil nil nil nil nil
Controle_de_l'optimiseur

Limite_de_temps(heures)............................. 2
Nombre_max_de_horaires_a_retenir.................... 5
Arreter_a_la_1ere_horaire_realisable(0:non,1:oui)... 0
Saut_d'integralite_max.(%_entre_0_et_100)........... 0
Plateau_de_la_relaxation_lineaire
nombre_d'iterations................................. 1
pourcentage_d'amelioration_min...................... 1
profondeur_max_pour_branchement_local............... 1

Periodes_de_quotas
Numero_de_la_periode  Debut  Fin
       1              07:30  07:30

Liste_des_quarts_de_travail
# # # # # # # #
JOUR    7:15   d     1   1   1    JOUR,SOIR          JOUR,NUIT,SOIR
NUIT    7:15   n     1   8   8    JOUR,NUIT,SOIR          JOUR,NUIT,SOIR
SOIR    7:15   e     1   8   8    SOIR          JOUR,NUIT,SOIR

Contraintes_de_quotas
# # # # # # # #
 1  JOUR    = 3  1  1  1    1999/10/03
 1  JOUR    = 3  1  1  1    1999/10/04
(...)
 1  NUIT    = 2  1  1  1    1999/10/03
 1  NUIT    = 2  1  1  1    1999/10/04
(...)
 1  SOIR    = 2  1  1  1    1999/10/03
 1  SOIR    = 2  1  1  1    1999/10/04

Liste_des_infirmieres
```

```
Identification
Numero_d'employe...   000000000903
Prenom_Nom....   xx
OIIQ        0
Titre_d'emploi...     1
Ancienneto...     6361

Info_horaire_courant
Quarts_acceptables.....  JOUR,NUIT,SOIR
Nb_max_d'affectations_dans_l'horizon..  20
Nb_min_d'affectations_dans_l'horizon..  20

Regles_actives(0:non,1:oui)
Fins_de_semaine.....  1
Conges/vacances_a_accorder.....  0
Non-presences_a_accorder_pour_l'hor
Jour_au_plus_tot..... nil
Jour_au_plus_tard..... nil
Nb_min_de_jours..... 0
Nb_max_de_jours..... 0
Jours_candidats..... nil

Valeurs_des_ratios_de_quarts
Ratio_max_de_quarts_de_jour.....  33
Ratio_min_de_quarts_de_jour.....  33
Ratio_max_de_quarts_de_soir.....  33
Ratio_min_de_quarts_de_soir.....  33
Ratio_max_de_quarts_de_nuit.....  33
Ratio_min_de_quarts_de_nuit.....  33

Affectations_consecutives
Nb_max_de_jours_consecutifs_conges....  4
Nb_min_de_jours_consecutifs_conges....  2
Nb_max_de_quarts_de_jour_consecutifs..  5
Nb_min_de_quarts_de_jour_consecutifs..  1
Nb_max_de_quarts_de_soir_consecutifs..  5
Nb_min_de_quarts_de_soir_consecutifs..  1
Nb_max_de_quarts_de_nuit_consecutifs..  5
Nb_min_de_quarts_de_nuit_consecutifs..  1
Nb_max_de_fs_consecutives_trav.....  1
Nb_min_de_fs_consecutive_trav.....  1
Nb_max_de_fs_consecutives_conge.....  1
```

```
Nb_min_de_fs_consecutive_congé.....  1

Charges_de_travail
Date_de_travail Date_de_fin Min.hr Max._heures_trav
1999/10/03    1999/10/09      36.25      36.25
1999/10/10    1999/10/16      36.25      36.25
1999/10/17    1999/10/23      36.25      36.25
1999/10/24    1999/10/30      36.25      36.25

Salaire
Valeur(x10)$   Jours   Quarts_de_travail
nil            nil     nil

Affectation_preferees
Valeur([-10,10)]  Jours    Quarts_de_travail
-5             Di          NUIT
-5             Lu          NUIT
-5             Ma          NUIT
-5             Me          NUIT
-5             Je          NUIT
-5             Ve          NUIT
-5             Sa          NUIT
-8   1999/10/05  NUIT
-8   1999/10/06  NUIT
-8   1999/10/07  NUIT

Pre_affectations
Jours  Quarts_de_travail
nil           nil

Affectations_interdites
Jours  Quarts_de_travail
nil           nil

Info_horaire_precedent
Jours_de_travail  Quarts_de_travail
nil           nil

Vacances_et_jours_feries_recus_durant
nil

Vecteur_de_ressource_initial
```

```
Fin_de_semaine  Rotation
    0               0

Affectation_precedent_la_premiere_affectation
Jours_de_travail  Quarts_de_travail
nil           nil

(...)
```